Research Product 90-31

Human-Computer Interaction in Tactical Operations: Designing for Effective Human-Computer Dialogue

September 1990

Fort Leavenworth Field Unit
Systems Research Laboratory

U.S. Army Research Institute for the Behavioral and Social Sciences

U.S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency Under the Jurisdiction of the Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

JON W. BLADES
COL, IN
Commanding

Research accomplished under contract for
the Department of the Army

VRC Corporation

Technical review by

Richard E. Maisano

NOTICES

~~**DISTRIBUTION**: Primary distribution of this report has been made by ARI. Please address correspondence concerning distribution of reports to: U.S. Army Research Institute for the Behavioral and Social Sciences, ATTN: PERI-POX, 500 Eisenhower Ave., Alexandria, Virginia 22333-5600.~~

FINAL DISPOSITION: This report may be destroyed when it is no longer needed. Please do not return it to the U.S. Army Research Institute for the Behavioral and Social Sciences.

NOTE: The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

UNCLASSIFIED

' CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

ORT SECURITY CLASSIFICATION ssified	1b. RESTRICTIVE MARKINGS --
JRITY CLASSIFICATION AUTHORITY LASSIFICATION/DOWNGRADING SCHEDULE	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
ORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) ARI Research Product 90-31
AE OF PERFORMING ORGANIZATION orporation — 6b. OFFICE SYMBOL (If applicable) --	7a. NAME OF MONITORING ORGANIZATION U.S. Army Research Institute Fort Leavenworth Field Unit
RESS (City, State, and ZIP Code) ng Court, Suite E and Oaks, CA 91360	7b. ADDRESS (City, State, and ZIP Code) P.O. Box 3407 Fort Leavenworth, KS 66027-0347
ME OF FUNDING/SPONSORING GANIZATION U.S. Army Research tute for the Behavioral ocial Sciences — 8b. OFFICE SYMBOL (If applicable) PERI-S	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MDA903-86-C-0210
RESS (City, State, and ZIP Code) Eisenhower Avenue ndria, VA 22333-5600	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 62785A; PROJECT NO. 790; TASK NO. 1304; WORK UNIT ACCESSION NO. C2

LE (Include Security Classification)
1-Computer Interaction in Tactical Operations: Designing for Effective 1-Computer Dialogue

SONAL AUTHOR(S)
nayer, Richard W. (VRC); and Fallesen, Jon J. (ARI)

PE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
l	FROM 88/01 TO 90/07	1990, September	

PLEMENTARY NOTATION

COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
LD	GROUP	SUB-GROUP	Command and control; Human factors; Battlefield automated systems; Command language; Human-computer dialogue (Continued)

TRACT (Continue on reverse if necessary and identify by block number)

This report presents guidelines for designing human-computer dialogue for tactical ations. Researchers consolidated sources of information into a form intended for use by gners of tactical computer systems to give them a basis to assess the military impact of ogue design and take leadership in improving the usability of future systems. This rt was developed to support dialogue design for two general situations: (1) the genera- of specifications for relatively large-scale systems in which the specific design and lopment will be performed by another, and (2) the development of relatively small-scale ial-purpose systems in which the reader will be the designer and developer, perhaps with aid of a programmer.

The user-computer dialogue is clearly the key to developing systems that fit in with 's goals and tasks. Consequently, this guide will emphasize the essence of the dialogue, ification of fundamental issues, performance of front-end analyses, selection between rnative dialogue types, and testing for usability. This guide will not address (at least (Continued)

TRIBUTION/AVAILABILITY OF ABSTRACT NCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
AME OF RESPONSIBLE INDIVIDUAL n J. Fallesen	22b TELEPHONE (Include Area Code) (913) 684-4933 — 22c. OFFICE SYMBOL PERI-SL

m 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED

ARI Research Product 90-31

18. SUBJECT TERMS (Continued)

Menu selection
Direct manipulation dialogue
Natural language dialogue
Form-fill dialogue
Tactical computers
User-computer interfaces
Guidelines

19. ABSTRACT (Continued)

to any major degree) the issues of data display, contents of on-line documentation and help, data transmission, hardware devices, or general human engineering considerations. For those interested in reading further, a reading list and a selected bibliography are provided.

Accession For		
NTIS GRA&I		☑
DTIC TAB		☐
Unannounced		☐
Justification		
By		
Distribution/		
Availability Codes		
Dist	Avail and/or Special	
A-1		

Research Product 90-31

Human-Computer Interaction in Tactical Operations: Designing for Effective Human-Computer Dialogue

Richard W. Obermayer
VRC Corporation

Jon J. Fallesen
U.S. Army Research Institute

Field Unit at Fort Leavenworth, Kansas
Stanley M. Halpin, Chief

Systems Research Laboratory
Robin L. Keesee, Director

U.S. Army Research Institute for the Behavioral and Social Sciences
5001 Eisenhower Avenue, Alexandria, Virginia 22333-5600

Office, Deputy Chief of Staff for Personnel
Department of the Army

September 1990

Army Project Number
2Q162785A790

Human Performance Effectiveness and Simulation

FOREWORD

The U.S. Army Research Institute for the Behavioral and Social Sciences (ARI) Field Unit at Fort Leavenworth conducts an extensive program of research in many aspects of command and control and soldier performance. One aspect of that research deals with developing tactical command and control systems and assessing user requirements. Through several years of interaction with the combat development community that works on tactical command and control systems, researchers have learned that the human-computer interface in tactical systems is dictated to a large extent by technologies available at the time. Too little front-end attention is given to determining the needs of the user for interacting with the system. These means of interaction, referred to as dialogue, should be considered and analyzed in communication terms.

Human-computer dialogue is the crucial aspect of a tactical computer system interface and a large factor in determining the success of operator use. A well-designed dialogue should contribute to user acceptance of the system and facilitate rapid training. Unnatural or stilted dialogues can impact negatively on the system and lead to suboptimal mission performance. These guidelines represent the best available information on selecting dialogue types and designing dialogue.

Edgar M. Johnson

EDGAR M. JOHNSON
Technical Director

HUMAN-COMPUTER INTERACTION IN TACTICAL OPERATIONS: DESIGNING FOR EFFECTIVE HUMAN-COMPUTER DIALOGUE

CONTENTS

LIST OF TABLES

LIST OF FIGURES

HUMAN-COMPUTER INTERACTION IN TACTICAL OPERATIONS: DESIGNING FOR EFFECTIVE HUMAN-COMPUTER DIALOGUE

Section 1. Introduction

The nature of human-computer interaction has changed dramatically over the past two decades. Initially, "users" produced decks of punched cards that were given to "operators"; one then waited for at least an hour or two for output that gave information for the next iteration. Now the interaction between user and computer may be relatively rapid and takes place directly through individual terminals, as for example, in modern tactical command and staff operations. This interaction may be viewed as a conversation or *dialogue* between user and computer.

This report will present guidelines for design of human-computer dialogue. Many sources ot information are consolidated into a form intended for combat developers, software developers, and applications programmers who have their own requirements definition. To give them a basis to (1) assess the operational impact of dialogue design and (2) take leadership in improving the usability of future systems.

The Promise of Computer Technology

Modern battlefield scenarios require the Army to fight out-numbered, strike deep and fast, and respond to narrow windows of opportunity. Modern computer technology can help to meet these challenges in many ways. For example, an analysis of the G3 section of U.S. Army corps and division main command posts (CAORA, 1985) identified fifty-three different G3 Main analytic opportunities for computer aiding based on criteria of importance and feasibility. Clearly, the U.S. Army Tactical Command and Control System is transitioning to a highly automated system.

Problem

Unfortunately, we have seen that computer technology can sometimes fail to yield expected benefits. For example, the following comments are representative of user responses for existing commercial systems (Nickerson, 1981):

- "The system was not designed with my job in mind."
- "Effective use of the system depends on knowing too many details."
- "The commands that I have to use in order to instruct the computer seem arbitrary."
- "Th names by which actions are identified are not descriptive; they are difficult to know and remember."
- "The need to be letter perfect in designating commands is frustrating."
- "I get confused among the languages and conventions of the various systems. A given control character may mean something in one system and something else in another."
- "I don't understand what's going on within the system."

An ARI User Acceptance Workshop (Reidel, 1988) listed a number of factors and causes of system non-acceptance, including:

- incompatible task representation,
- unfamiliar procedures,
- distrust of system builders ,
- does not "speak" user's language,
- inadequate training,
- loss of control over work,
- increased workload, and
- bad interface design.

In short, although powerful computer tools can be crafted, there is the possibility that some features may not be usable by the intended system users.

Affecting System Design

Furthermore, we find that it is difficult to get usable systems designed with the user and the user's needs in mind. For example, it has been found (Meister & Sullivan, 1967; Meister, Sullivan & Finley, 1969) that manuals, handbooks and guidelines were not used by designers. It was suggested that the user considerations be included explicitly in the system specification to insure appropriate consideration by the designers. Possibly, the "specifier" must assume more of the functions of the "designer."

If usability deficiencies are noted during the design process, often only "band aid" or cosmetic solutions are possible (Rouse and Boff, 1987). Unfortunately, many decisions that are subtly important to system use are made early in the developmental process. Consequently, the design process must be iterative and involve testing with representative users.

Purpose

Military subject matter experts can provide the leadership needed to field the automation that will meet current and future challenges. Subject matter experts, who have a deep understanding of the underlying tasks and requirements, can have a beneficial impact on dialogue design. Therefore, this report is intended to provide such individuals with information and tools to influence future dialogue design.

Scope

This report was developed to support dialogue design for two general situations: (1) the generation of specifications for relatively large-scale systems, in which the specific design and development will be performed by another (e.g., a contractor), and (2) the development of relatively small-scale special-purpose systems in which the reader will be the designer and developer, perhaps with the aid of a programmer (e.g., software such as that available through C2MUG [Command and Control Microcomputer Users' Group].

The user-computer dialogue is clearly the key to developing systems that fit in with the user's goals and tasks. Consequently, this guide will emphasize the essence of the dialogue, clarification of fundamental issues, performance of front-end analyses, selection between alternative dialogue types, and testing for usability. This guide will *not* address (at least to any major degree) the issues of data display, contents of on-line documentation and helps, data transmission, hardware devices, or general human engineering considerations. Nevertheless, in an integrated design effort focussed on developing a usable system, all of these need to be developed in parallel.

We recommend, at least for items critical to the desired application, that the reader refer to existing references for information to supplement that presented here (see the reading list in Appendix A and the selected bibliography in Appendix B).

Section 2. User-Computer Interface Design Processes

Analysis, design and testing of the dialogue between the user and computer occurs in the context of the design of the total user-computer interface (UCI). This section will briefly review the total UCI design process to establish a framework for the remainder of this document.

Military-System User-Computer Interface Design Processes

The military handbook DOD-HDBK-761 (DOD, 1985) presents human engineering guidelines for management information systems. The process for analyzing, designing and testing UCI designs, as added to the revised DOD-HDBK-761 (Baker, Eike, Malone and Peterson, 1988), is depicted in Figure 2-1. The UCI design process is divided into three phases, consisting of a total of ten steps.

Phase I. Requirements analysis. The first phase involves planning, analysis of user needs, and the preparation of a functional specification for the UCI.

These activities are conducted during the System Concept Development and System Requirements Analysis phases of the Army's Materiel Acquisition Process (MAP). The UCI functional specification serves as an input to the System Segment Specification, the Operational Concept Document and the Preliminary Interface Requirements Specification.

Phase II. UCI design and development. While the functional specification of Phase I addresses "what" the UCI is to do, the second phase addresses design concepts and criteria for "how" the recommended user-computer interface is to work.

The UCI design activities proceed in parallel with the preliminary and detailed design phases of the software development process. The overall UCI design concept is developed in conjunction with the MAP Preliminary Design Phase and the Preliminary Design Review; the UCI design concepts are developed in conjunction with the MAP Detailed Design Phase, and are assessed in the Critical Design Review.

Phase III. UCI test and integration. The object of this phase is to evaluate the UCI design, complete the integration of the UCI with system software, and produce the UCI implementation specification.

The activities of the third phase are conducted in the system test and integration phases of software development. The UCI implementation specification serves as an input to the software description, operations and support documents, system integration test procedures, and the software product specification.

Relationship to this document. As shown on the right-hand side of Figure 2-1, this document supports the UCI design process.

Section 3, provides concepts and theory which may affect the way the designer views the user-computer dialogue design problem:

- The designer may view the user-computer interaction from a systems perspective, a dialogue-partner perspective, a tool perspective, and a communications media perspective.

- The designer may consider a number of alternative configurations of users and computers.

- The designer may view dialogue design as the creation of a language for user-computer communication. It is useful to consider the dialogue in terms of the linguistic concepts of semantical, syntactical, and lexical levels, as well as spatial layout of specific input and output devices.

- The designer will be required to select from a classification of dialogue types.

- The designer should consider a range of user-centered criteria during the design and testing process.

Section 4, presents methods for analyses that will be useful during the Phase I activity shown in Figure 2-1:

- These analyses are suited for identifying dialogue design information, and are keyed to the multiple dialogue levels discussed in Section 3.

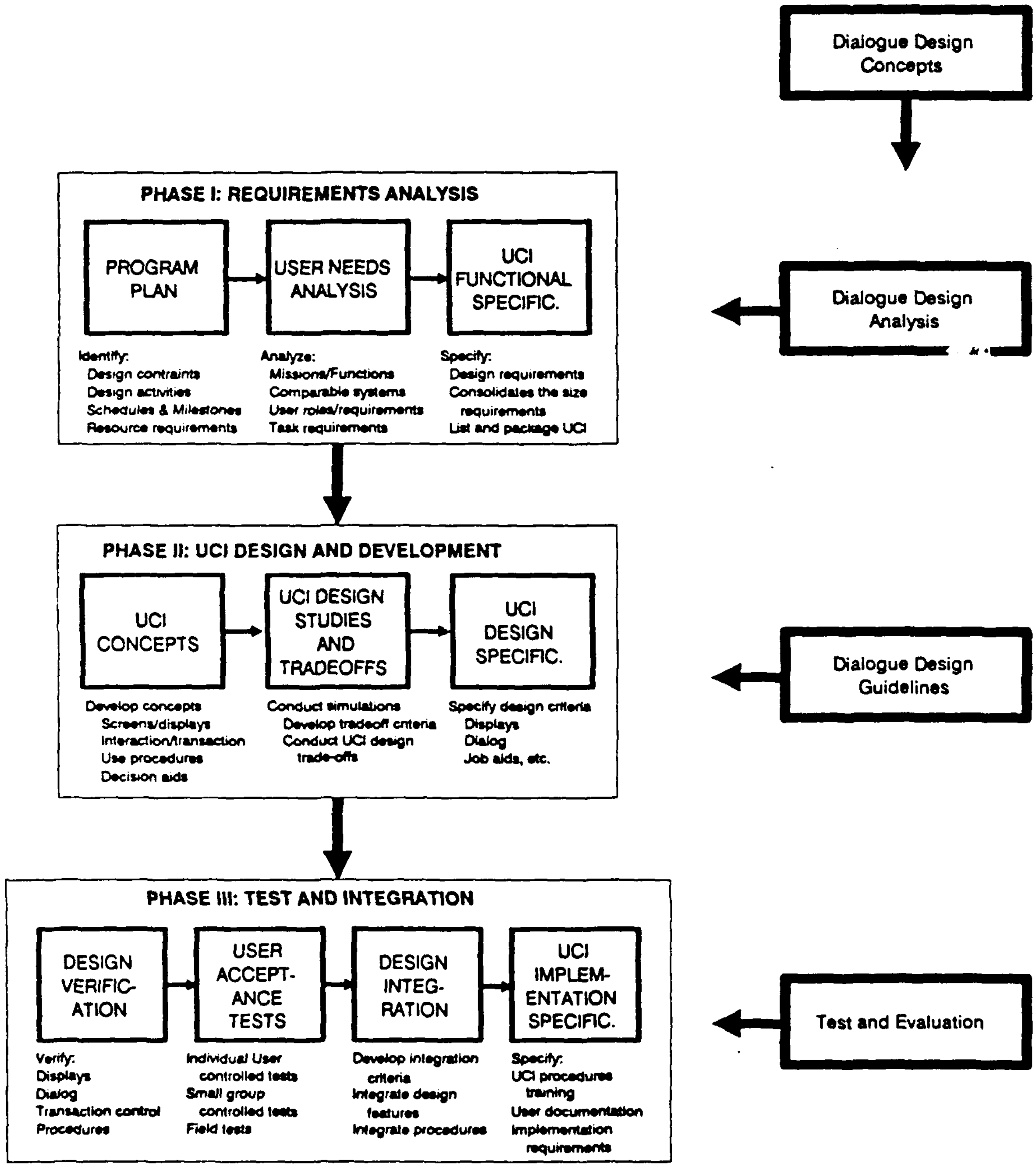
- These analyses are intended to augment the other general UCI analyses for mission, function, and task analyses indicated for Phase I in Figure 2-1.

Section 5, presents alternatives, guidelines, and criteria for dialogue design to support the dialogue aspects of UCI concepts and design studies shown in Figure 2-1:

- The advantages and disadvantages of each dialogue type and desirable combinations are discussed.

- Examples and recommendations are presented to aid design decision making.

Section 6 presents information which should be useful for dialogue design test and evaluation. Specific types of tests and measurement are discussed to support the Phase III activity shown in Figure 2-1.



a. DoD HDBK 761.

b. These Guidelines.

Figure 2-1. User-computer interaction design process.

Iterative User-Centered Design Processes

A criticism of many human-machine systems is that they are not usable; that is, while these systems may have be able to perform useful functions, the users are often unable to produce the desired results. Such systems often are not well accepted by the users. A philosophy for design, advocated by Gould (1988) and many others, has had growing acceptance over the past ten years. This process consists of four basic principles:

- early focus on users, achieved by designers having direct contact with users through interviews, surveys and user participation in design, rather that basing design on the results of formal analyses,
- integrated design, in which the interface, helps, training, and documentation are developed in parallel,
- continual user testing, using simulations and prototypes to measure, user performance and reactions are measured throughout the design process, and
- iterative design, provision is made to revise the design and repeat the three steps of focus on the user, integrated design, and user testing.

There have been a number of successful applications of these principles and the ideas are considered commonplace today. Many are convinced that these principles are the required basis for the design of systems which will be useful, easy to learn, usable by the intended users, and liked by the users.

While this approach has been, in general, well received, there has been some organizational resistance, and it is clear that most computer systems and applications are not developed in accordance with this process. For example, it can be argued that:

- there is not enough time or money for iterative design,
- design should be "right" the first time,
- users don't know what they want,
- usability cannot be measured,
- small changes to complex software have far-reaching effects, and
- it is too difficult to get recommendations implemented.

Relationship to this document. It is not the purpose of this document to recommend a specific design process. While the goal of designing for usability is certainly desirable, it is not clear that informal incremental optimization is always better than formal analysis-specify-design-test methods. Readers involved in development of large-system development may have little choice but to follow formal design procedures. On the other hand, readers involved in development of small-scale systems can consider using less formal procedures involving continual iteration and testing with users.

One approach to reduce organizational resistance to iterative design (based on time and cost), is to provide better software tools for the design of the user-computer interface. Further, there has been some emphasis on the separation between *style* and *content* in user-computer interface design. That is, one can separate such issues as whether to use menu-selection from issues associated with the design of application software with appropriate capabilities. Often, the user-computer interface comprises the majority of the total system software. Therefore, significant savings can be achieved by developing re-usable interface software. Furthermore, steps can be taken toward standardization, and the specifier may be able to take on an increased role as an implementor.

As shown in Figure 2-2, this document can be used for informal iterative design procedures. Section 4 provides analytic techniques which emphasize analysis of user requirements and functional requirements, that is, analysis directed at identifying the *content* of the interface. Section 5 provides information which is primarily related to user-computer dialogue *style* design. Section 6 provides testing methods appropriate for rapid prototyping and testing which can be applied continuously throughout the design process.

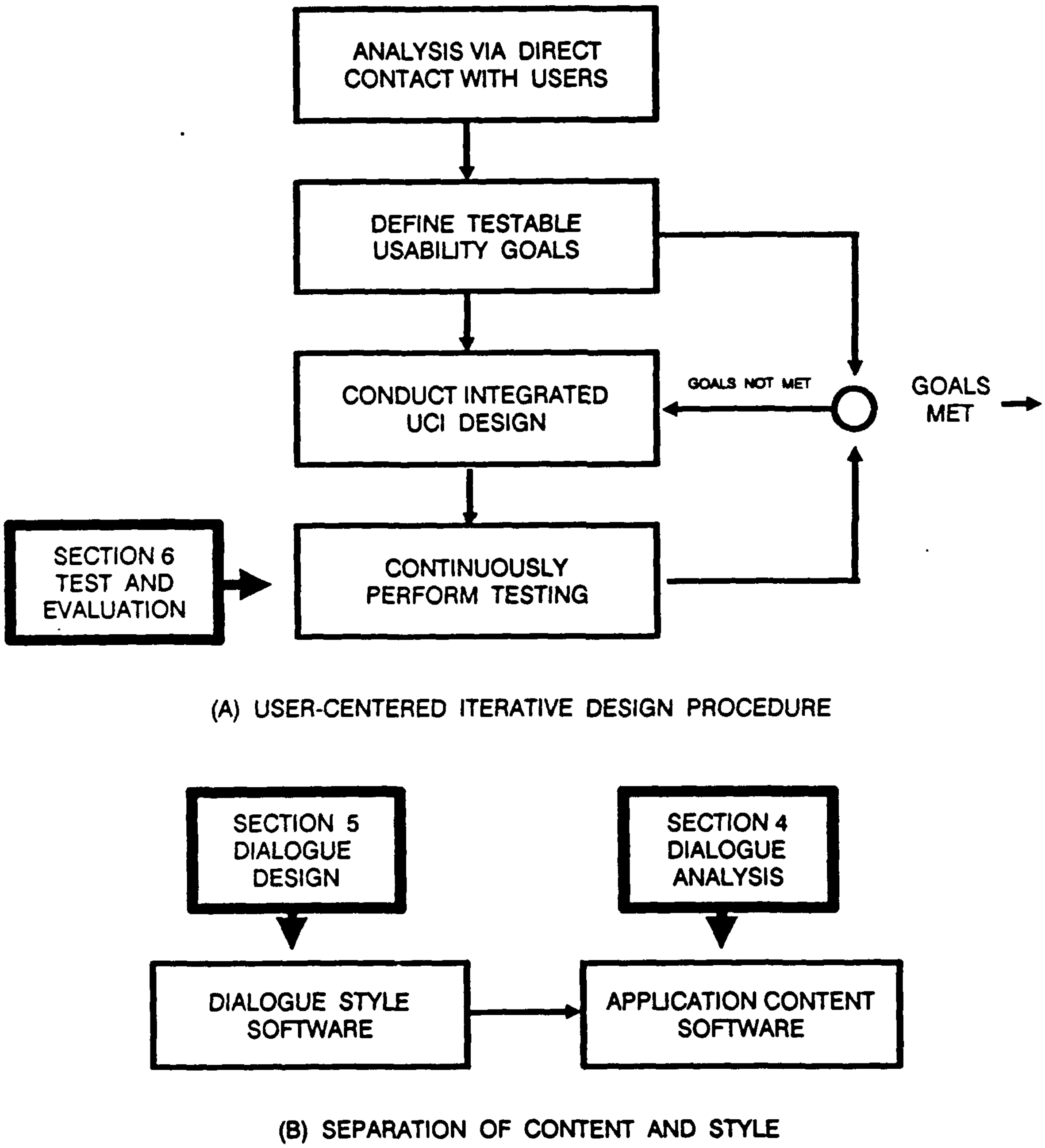


Figure 2-2. Iterative user-centered design process.

Section 3. Dialogue Design Concepts

Purpose

This section defines general concepts to the reader for subsequent sections on analysis, specification, design, and evaluation.

Alternative Views of Human-Computer Interaction

Human-computer interaction can be viewed from at least four different perspectives (Kammersgaard, 1988), as portrayed in Figure 3-1.

The systems perspective. In this perspective, a system is viewed as consisting of different components that have similar properties:

- all components are characterized by a set of data types and set of actions,
- components can transfer data to each other, and
- data are processed according to predefined rules.

Human-computer interaction as seen from a systems perspective deals with the exchange of data between a human and an automatic component of a system. The essential quality of a user interface is to make sure that the transmission of data between the human and the automatic components takes place according to predefined rules.

From this perspective, design stresses the timeliness and accuracy of the performance of each task, the effect of errors, and the transformation of information and the compatibility of information produced by one task as the input for the next task.

The tool perspective. From the tool perspective, the computer system is seen as providing the user with a tool-kit which is expected to be useful in accomplishing the user's tasks. The user is seen as a person who has skills relevant within the domain, and the development of computer-based tools assumes that the tools are to be employed by skilled users.

Choosing a tool, using it, and evaluating the result of its use is a typical sequence which is repeated over and over. The user must be able to select and apply each tool to achieve a variety of products depending on the available *functionality* (the set of system features available to the user). For example, word processing tools are used to produce reports, and computer drawing tools are used to produce block diagrams or art work.

The dialogue partner perspective. Humans and computers can be regarded as partners in a dialogue. User and computer can both act as sender and receiver in the communication process to accomplish desired tasks. A possible extension of this viewpoint is to attempt to produce a user interface in which the computer application acts like a human in a communication process. Humans do not always communicate in an optimal way, so the designer wants to use only those styles which increase usability.

Technological development has progressed from physical tools which magnify man's physical and perceptual capacity, to knowledge and cognitive tools based on artificial intelligence techniques. Expert systems are an example of this technology, in which a set of rules

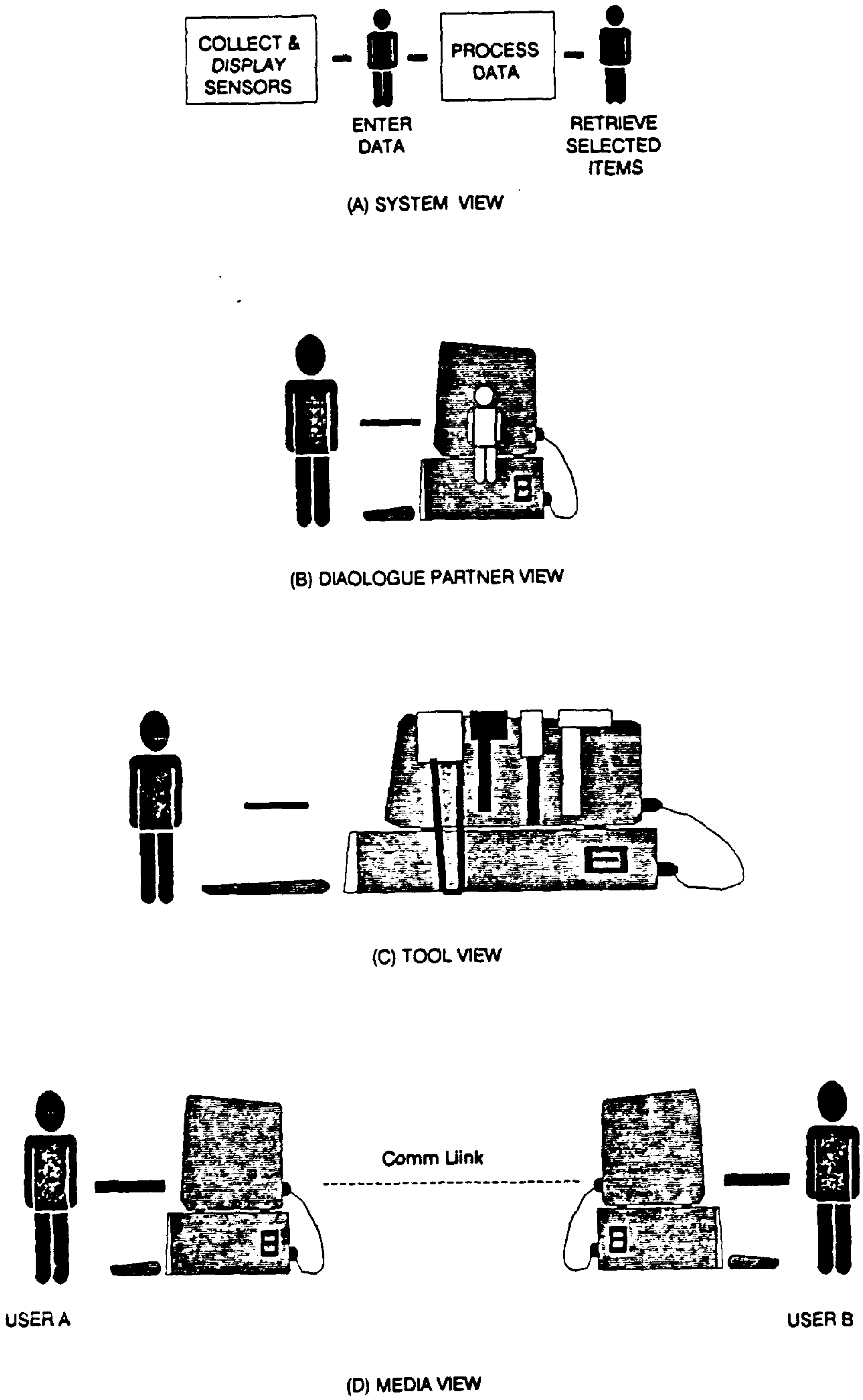


Figure 3-1. Four views of human-computer interaction.

is used to make inferences that correspond to a consultation that could occur with a human expert.

An encounter with an "intelligent" computer consultant could consist of the following:

- user initiates a session,
- computer controls data gathering,
- computer offers a solution,
- user may ask for an explanation, and
- user accepts or overrides the computer's solution.

Cognitive tools based on artificial intelligence techniques can be viewed as *instruments* (which magnify the user's capability) or *protheses* (which supplement the user's deficiencies). These tools create new challenges for combining human intelligence and machine power into an effective integrated system (cf., Mancini, Woods, & Hollnagel, 1987).

Artificial intelligence techniques may permit software architectures which can support truly interactive dialogue between user and computer, that is, an "intelligent" interface (Halpin, 1984). In such a system, commands and requests would be interpreted based on an understanding of the user's characteristics and goals, and of the current state of the world. The software would use constantly updated models to transmit relevant information to the user in an appropriate way. In the design of such a system, in addition to consideration of the knowledge the user requires in an interaction, the designer must also consider the knowledge the system requires.

The media perspective. From t[illegible] media perspective, the computer is seen as a medium through which humans communic[illegible] ith each other. Of course, there must be more than one user for this view to be taken. The media perspective requires that the designer focus on the language aspects of the use of computers. Depending on hardware and software capability, messages may be of various types (text, graphics, voice), in real time (synchronous) or delayed (asynchronous), and at the same (face-to-face) or remote locations. Note that two levels of dialogue may be involved, i.e. user-user dialogue conducted with an underlying user-computer dialogue at each end of the communication channel. In addition to providing a communication channel, the combined computers may promote and augment cooperative shared work among multiple users (called *computer-supported cooperative work*).

It may be useful to employ any or all of these views during the design and development of a system to ensure comprehensive treatment.

Types of Dialogue Configurations

Often human-computer interaction deals with, or assumes, a configuration with one user and one computer. However, this is not the only configuration, as may be seen in Figure 3-2. In some cases, close examination may reveal hidden additional human and computer interaction; for example, the dialogue may include other individuals who provide assistance or supervision.

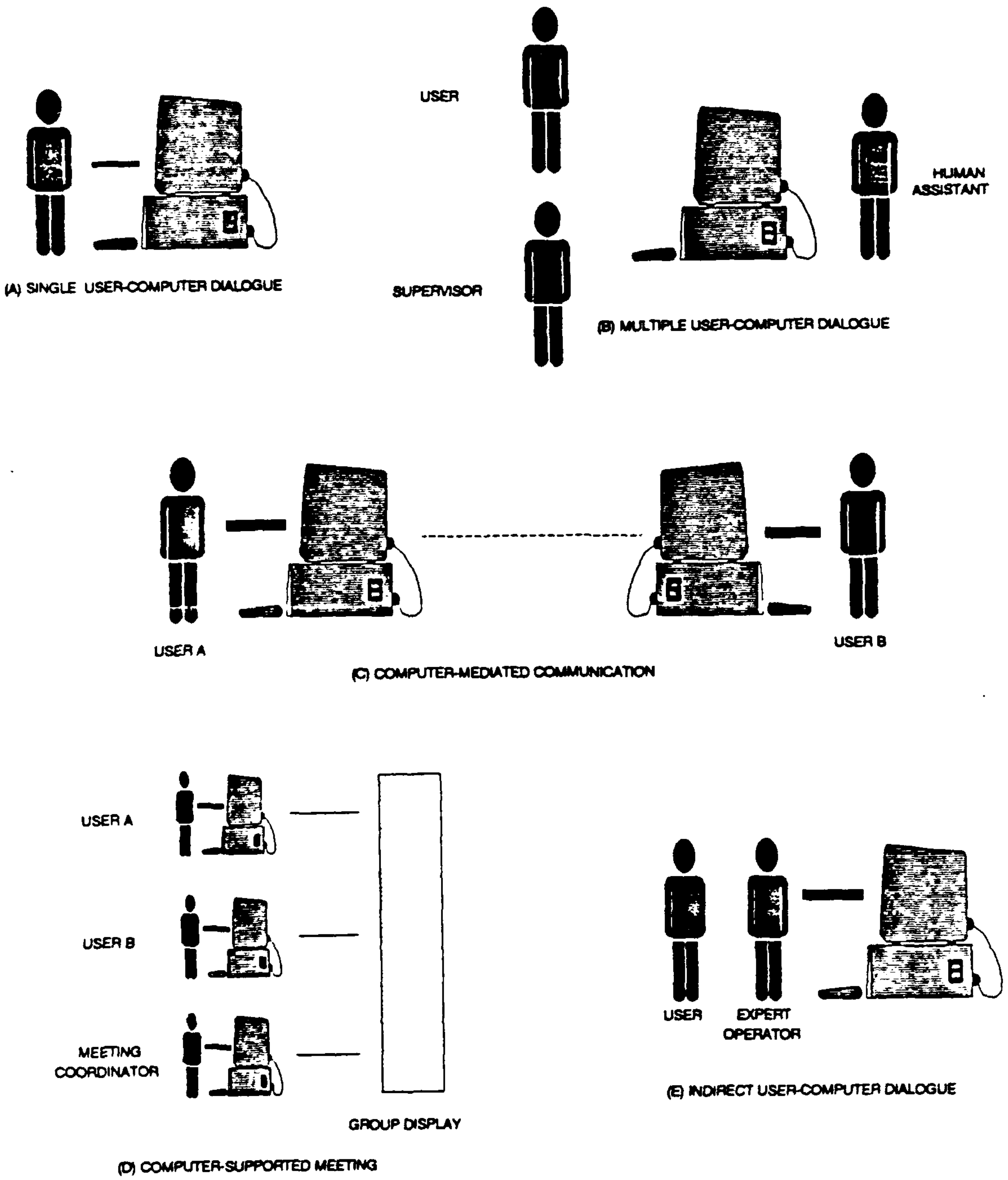


Figure 3-2. Alternative human-computer configurations.

Additionally, when the computer can be a media for communication, the design must deal with computer-mediated person-person dialogue. Furthermore, computers can be used in meeting rooms or command centers, so that dialogue design must consider multiple users.

Levels of Dialogue

The development of a usable human-computer dialogue must deal with a hierarchy of design users, as depicted in Figure 3-3. The hierarchy of design issues is discussed in the following paragraphs.

User types and tasks. Users differ in knowledge, skills and backgrounds. The design should focus on the user so that the dialogue can be tailored to the user and attempt to avoid potential confusions and mis-communications.

Each computer function must be specified to fit each task requirement. That is, the system must at least provide the necessary set of functions, and in a way that the user can use effectively. The object of the human-computer dialogue is the performance of these tasks and the accomplishment of a mission.

Semantic. A computer system is designed around a set of objects and the manipulation of these objects according to the user's needs. These are data structures and procedures in the system; but to the user they are conceptual entities and conceptual operations on these entities.

The semantic level deals with the meaning of the dialogue to the user. The user has a "point of view" or a "mental model" which provides a context for conversation. As with human-human communication, the contexts of the parties involved in a conversation should agree, or transmission of correct information may not occur.

As shown in Figure 3-4, the user's mental model is termed the USER MODEL (Norman and Draper, 1986, p. 47). The programmer's mental model used to create the system software is termed the DESIGN MODEL. Furthermore, the user is given information at the system-user interface which defines a SYSTEM IMAGE MODEL.

A general design goal is to achieve a common interpretation among these models. If the DESIGN MODEL and USER MODEL are identical the user should have no difficulty with dialogue meaning. However, the user may be able to adapt to a different or augmented model if it is accurately and completely reflected in the SYSTEM IMAGE MODEL. Unfortunately, some systems have been designed so that the SYSTEM IMAGE MODEL was different or simpler than the actual system operation, leading to user confusion.

Desirable goals at this level of dialogue design are semantic consistency (i.e., identical portions of the dialogue always mean the same thing) throughout the system design, and development of a vocabulary for communication which is unique and clearly differentiated (i.e., command terms and abbreviations are clearly associated with task requirements by the user).

Syntactical. A user communicates with a computer system in terms of a language structure built from a few elements: commands, arguments, context and state variables. The syntax is the order, combinations, and punctuation which are legal statements acceptable to the computer. The conceptual model is embedded in the language in the meaning of each command, while the syntactic level determines the elements of the command coding.

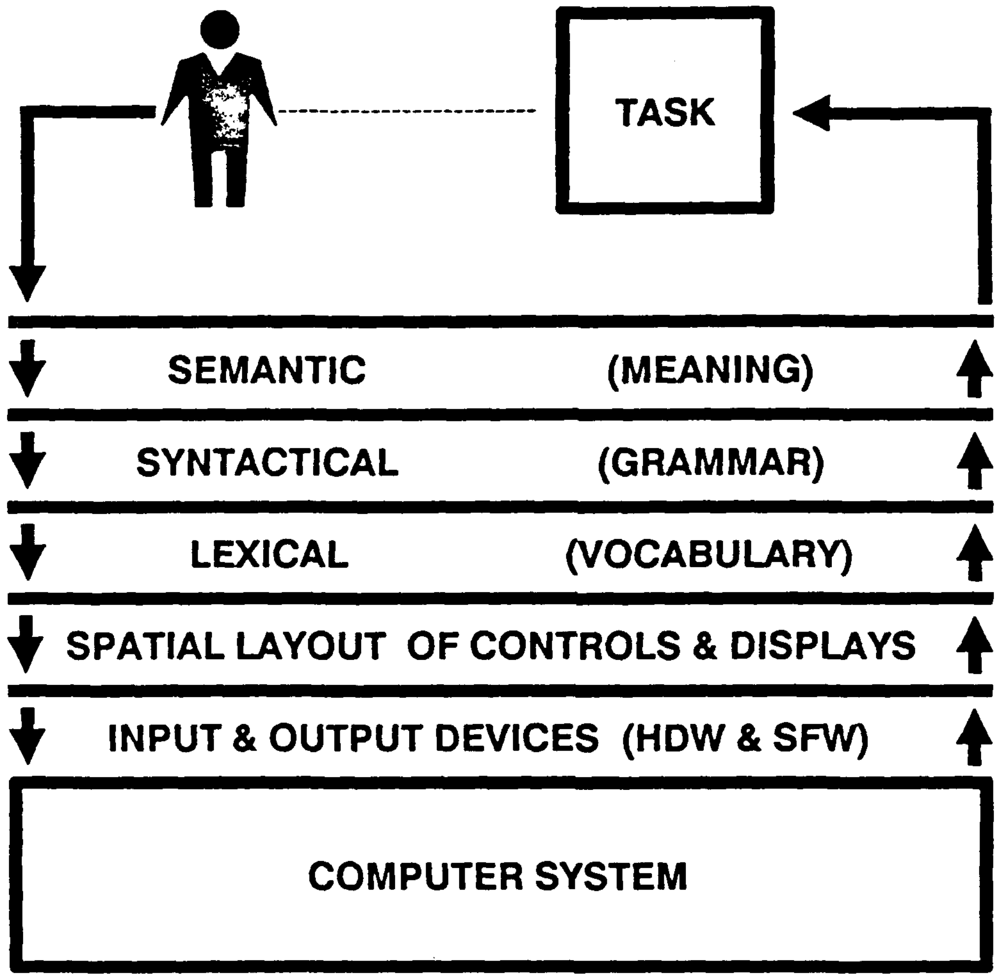


Figure 3-3. Stratified human-computer dialogues.

The dialogue design has direct impact on system design at the syntactic level, for at this level, interpretation and translation of the language occurs. Design goals, at this level, are to ensure that a complete set of functions are provided, and that the syntax is consistent (i.e., all similar functions are represented in the same way).

Lexical. A lexicon is a dictionary of the smallest elements of a language which can convey meaning, i.e., the vocabulary of a language distinct from its grammar.

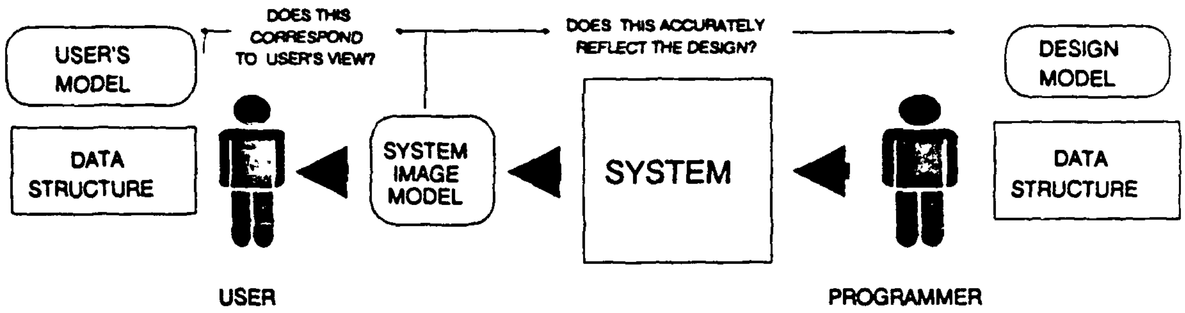


Figure 3-4. User, system image and design models.

At the lexical level, the user-system dialogue is expressed as primitive language elements produced by specific input or output device manipulation, for example, letters of the alphabet, special characters, Return, and Control. Additional consideration must be given to features associated with system control, such as the use of windows to timeshare the display area among multiple tasks (and the means to select, locate, size, scroll each window) and the manner in which menus are selected (e.g., pull-down menus, navigation through the menu structure, and selection of items from lists).

Spatial layout. An important aspect of the physical interface between the user and system is determined by the arrangement of the input/output devices, including the display screen layout.

Physical devices. The characteristics of the physical devices, e.g., screen resolution and keyboard tactile feedback, are the lowest level in the user-computer dialogue.

Types of Dialogues

Quite long lists of alternative types of dialogues can be formed. For example, Martin (1973) lists the following types of dialogues: programming language, English language, limited English language, question and answer, dialogue with mnemonics, dialogue with programming-like statements, computer initiated, form-filling, menu select, built into hardware, dialogue with pointing device, fixed-panel response, modifiable-panel response, graphics with symbolic manipulation, graphics with chart displays, graphics with photo frames, voice answerback, and third party. Clearly many variations are possible.

A small set of dialogue types, discussed by Shneiderman (1986), is believed to be sufficient to present a set of design guidelines which the reader can extend to military applications. These are:

(1) Menu selection. Dialogue in which the user selects an alternative from a list of options.

(2) Form fill-in. Dialogue in which the user enters data into a blank form.

(3) Command language. Dialogue in which the user enters commands and arguments to initiate computer action.

(4) Natural language. Dialogue in which the user uses ordinary (english) language. For any near term application, the language will be a restricted form of natural language.

(5) Direct manipulation. Dialogue in which the user modifies a model presented on the computer screen, indicates changes to the state of the model, and thereby directs computer actions.

Each of these dialogue types will be discussed in more detail in Section 5 of this document, along with the notion of combining dialogue types into a hybrid.

User Dialogue Criteria

To achieve high levels of user acceptance, system design should consider the criteria that a user would consider important. Simes and Sirsky (1985) developed the set of evaluation dimensions shown in Table 3-1 (only those directly related with dialogue issues are presented).

Table 3-1

Psychological Factors Related to Human-Computer Dialogue

Criterion	Definition
Tailorability	The capability for the user to modify, redefine or choose the form, structure, or type of dialogue displayed. The capability to redefine or specify names, abbreviations, or sequences of commands. User control of the volume, speed, and rate at which is presented or entered. The capability for the user to execute commands, functions, or processes in a fast and efficient manner. The use of commands or hardware that reduces the time required to accomplish the task.
Types of dialogues	The structure of dialogue techniques used for interaction between humans and computers. Some of these dialogues are independent of specific technology while others are restricted to monitor screens. The designer should note the type(s) of dialogues offered and the "appropriateness" of these dialogues for the tasks and functions performed.
Translation	The ability to recognize, translate, or interpret the user's input into the form needed by the system. The tolerance to interpret lower case when upper case is required, or to translate user entered data from one form to another. The ability to translate abbreviations, identify versions, and interpret terminology based on the user viewpoint.
User overrides	The capability for the user to correct, change, or override a system standard. For example, the user should be able to bypass or go around a sequential data entry scheme, cancel or undo the last action, or set of actions, or override validation checks on entered data. The user is the ultimate source for decisions. The user can be informed that a particular action or input may be in error; but the user should not be prevented from continuing.
Command uniformity	The user must be assured that every command, keyword, clause, and default will produce the same results and have the same meaning each time it is used. Two different commands should not perform the same sequence of operations, although two different sequences of commands might produce the same results. Further, the syntax of all commands should be the same. Position and order of arguments and special characters used as flags should be uniform between commands. Defaults should be uniform between commands and consistent with the system standard.

Section 4. Dialogue Design Analysis

This section defines analyses which yield information about *what* the human-computer system is to do, and requirements for dialogue design. The subsequent section (Section 5) will deal with *how*, specifically, the dialogue is to be implemented. In other words, this section treats dialogue *content*, while the following section treats dialogue *style*. Even if there was a standard set of interface software which removed dialogue style issues from consideration, the analyses in this section would still be necessary.

Overview of the Dialogue Analysis

It is safe to say that the quality of dialogue design will depend in large measure on the amount of detailed information available defining the application and the dialogue goals. In particular, it is necessary to achieve an understanding of the user task requirements; the alternative is the "common-sense" approach to dialogue design, which is risky.

The approach is user-centered and focuses on function and task analysis (perhaps the most-applied human factors analytic technique) since the analysis must be driven by the needs of the user to perform required tasks. The analysis proceeds through successively-detailed levels of the dialogue, following the structure which is dictated by viewing dialogue design as language development (e.g., semantic, syntactic, and lexicon levels), super-imposed on standard techniques for human engineering a workstation (e.g., layout and screen design). Each step in the analysis is presented in the same format: purpose, structure, and design relevance.

We recommend that all of these analyses be performed for any application, with the level of detail expanding as the design progresses through successive iterations. In addition to military application knowledge, hardware and software knowledge, these analyses frequently require behavioral science knowledge. Appropriate expertise in human factors and interface usability should be sought throughout the analytic effort.

Function and Task Analysis

Purpose. The purpose of the function and task analysis is (1) to identify each instance where human-computer dialogue may occur during the course of a mission for which the system may be used, and (2) to describe each transaction between user and computer in sufficient detail to enable subsequent analyses, decisions, and tests.

The function and task analysis may serve many purposes in a large-scale system design other than the dialogue design, including allocation of functions among users and automation, information display design, and information for development of manuals and training programs. Users of existing systems, even if systems are predominantly manual, are a necessary source of information.

Structure. The function and task analysis may be a combination of graphical and tabular methods (cf., Phillips, et al., 1988). A flow diagram of functions and tasks is recommended, showing the response of these tasks to external or synchronous events, procedures, and interactions with other personnel. The goal is to show meaningful units of work in terms of identifiable human-computer interactions. This is then coupled with a breakdown of task behaviors, required human capabilities, information display and human response requirements, and other information which can be used to assess design tradeoffs.

A number of graphical techniques have been used for function and task analysis. For example, IDEF0 (Softech, 1978) is a flow diagram technique, which in addition to inputs (left side of box) and outputs (right side of box) shows controls (top of box) and mechanisms (bottom of box). An example based on a division collection plan is shown in Figure 4-1. When developed to sufficient depth, each interaction is identified and its relation to mission accomplishment is evident.

Although task analytic techniques have been used extensively since first introduced (Miller, 1956), these methods have not been used heavily for systems which are stimulus-intensive, non-sequential and which involve mostly cognitive responses. Methods which do address these special applications are proprietary. However, Phillips, Bashinski, Ammerman & Fligg (1988) propose a technique for task analysis for dialogue design which parallels the content of this Section. Kincade and Anderson (1984) used the following guidelines for nuclear power plant control room analysis:

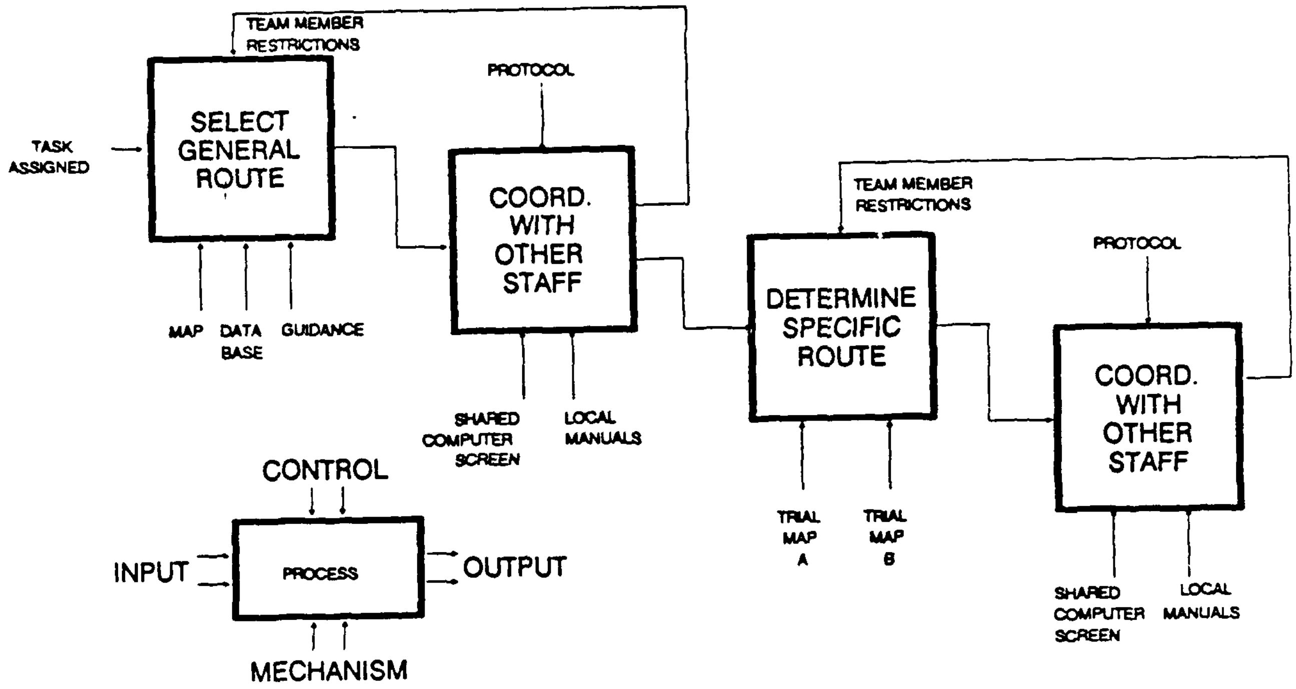
- Describe major task actions - interfaces involved, task sequences or patterns, impact, frequency, and importance of each action; and a timeline of actions.
- Describe task behaviors - behavioral description of the user-machine interaction.
- Assess human capabilities - human performance capabilities for sensing, associating, interpreting, and responding; identify features that can reduce task demands.
- State information and response requirements - information needed to detect, match, select, actuate and verify; response characteristics required.

For military tasks, function and task analyses can be described as a database structure (e.g., dBASE software application) which permits the entry of specific information throughout a hierarchical task structure. The structure should include:

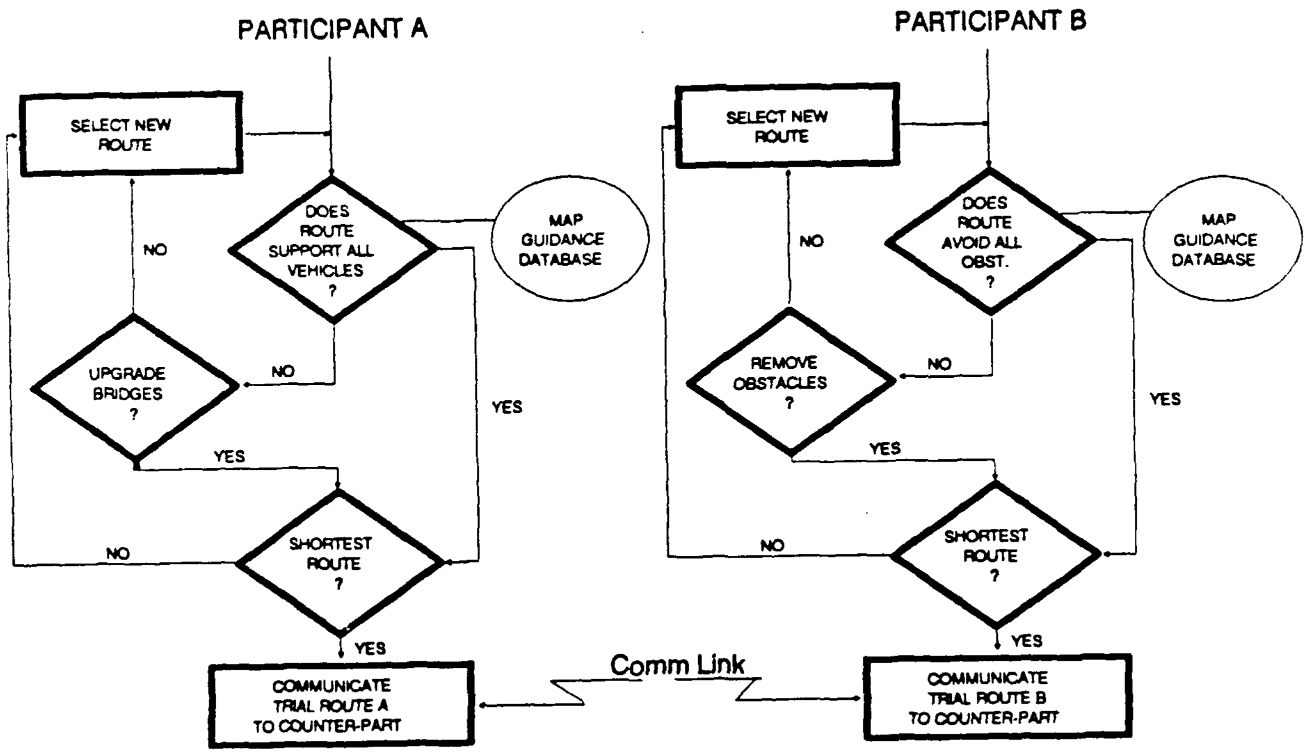
- mission breakdown - e.g., scenarios, missions, phases, segments,
- function and task breakdown - e.g., function, task, task element,
- design information for each task element - e.g., element number, element name, type, interface used, user performing task, stimulus (display) information to do task, responses required, type data entered, feedback provided to user, performance standards, response time (min, max, av), task workload (visual, cognitive, psychomotor), and task criticality, (see Kincade and Anderson analysis content, above),
- description of contingencies (emergencies) to be considered - e.g., name, type, consequences, cues, options, responses, and,
- alternative equipment configurations - e.g., interfaces, systems, equipment.

Design relevance. This analysis will provide a checklist of dialogue actions which will be used for checking dialogue completeness and for test criteria. The analysis forces the design team to examine the entire spectrum of use and consider the design issues which are posed.

The computer data base which can be produced will be useful for many subsequent analyses, e.g., identifying how often dialogue options are selected, allowing most frequent



(A) IDEF GRAPHIC MODEL



(B) TASK FLOW DIAGRAM (EXPANSION OF PORTION OF (A))

Figure 4-1. Example function and task representation methods.

selections to be placed at the top of a menu. The time interval between user encounters with the computer system, or use of specific system features, should be noted as this will have a major impact on the selection of dialogue modes.

User Analysis

Purpose. User-computer system design should focus on the user-computer partnership, for ultimately performance depends on how well the system is used and how well the system provides for user needs. Therefore:

- know the user, and
- focus design on the user.

Structure. Shneiderman (1987) advises that the following user characteristics be identified and considered for design impact: age, education, cultural/ethnic background, job experience, motivation, goals, personality, task-specific skills and abilities. The military user will be described in terms of MOS ratings, including operators and maintainers of many types.

Primary considerations in previous studies include (1) computer novice *vs* computer expert, (2) expert in task *vs* novice in task, (3) casual user *vs* frequent user, and (3) skill in typing and willing to type *vs* unskilled in typing and unwilling to type.

The designer for a computer system to be used in the command and staff environment is probably dealing with a relatively narrow range of users. (We assume that the users in the command and staff environment will be (1) mostly male, but could be either gender, (2) in the age range 19 to 50, (3) have a high school education or more, (4) have at least some typing skill, and (5) probably have some experience using computers.) Of course, the designer may have to deal with specific exceptions to this stereotype of users.

Early focus on users is commonly recommended for user-computer dialogue design. For example, Gould and Lewis (1985), recommend the following activities:

- talk with users,
- visit customer locations,
- observe users working,
- video tape users working,
- study the work organization,
- have workers think aloud while working,
- try it yourself,
- involve the user in the design process,
- administer surveys and questionnaires,
- observe users on competitive systems,

- develop printed or video scenarios
- develop early user manuals, and
- identify testable behavioral target goals.

Design relevance. Consideration of the user through preliminary design analyses will help to create a user-centered design. However, it is also wise to involve the user in the design process, and use of representative users will bring user characteristics under examination. In this way, considerations of specific user styles or task-specific factors will ultimately be included in the design.

Semantic Analysis

Purpose. A semantic analysis will attempt to reveal details of the way a user views the human-computer interaction (mental model). The structure and content of the user's knowledge and any preconceived notions of data structure should be documented during this analysis. This analysis should determine the meaning, priority and properties associated with the dialogue entities.

Structure. Mark (1986) points out that artificial intelligence techniques for knowledge representation can be used to identify the user's model. Although the user's model is formed and continually refined through contact with the system image model, one can attempt to identify features of the user's model which exist before design, and identify those features which it is unwise to attempt to modify. For example, if users' have years of experience with a manual filing system, it may be wise to retain the data structure of the manual file records. On the other hand, since data will be automatically retrieved, it is probably unnecessary to mimic the old physical locations (e.,g., aisles and stacks).

One form of knowledge-based representation defines terms by relating them to known terms. As shown in Figure 3-2, the arrows indicate a "kind-of" relationship, with all "things" in a computer system being divided into "objects" and "actions", and each is then progressively divided to form a kind-of hierarchy. Although not shown in Figure 3-2, it may be required that the language include modifiers, such as "properties" for each of the "objects".

The semantic map shown in Figure 4-2 can be derived directly from the function and task analysis (cf., Phillips, et al., 1988, p. 851). This can be done formally by adding additional items to the analysis of each task element corresponding to the objects, operations and properties involved in each task. For example:

Task Element	Object	Properties	Operations
...	...	...	...

It will be helpful later in the dialogue design process to include an estimation of the frequency (e.g., low, medium, high) and the priority (e.g., low, medium, high) for each semantic entity. For example:

Operation	Frequency	Priority
...	...	...

The programmer may add actions and objects to the system model not included in the model of a user who has not been exposed to the system. For example, a typist accustomed to a conventional typewriter will not know functions found in current word processor software, such as "block delete", and "retrieve file". The designer should carefully monitor any conflicts or basic changes to the original model.

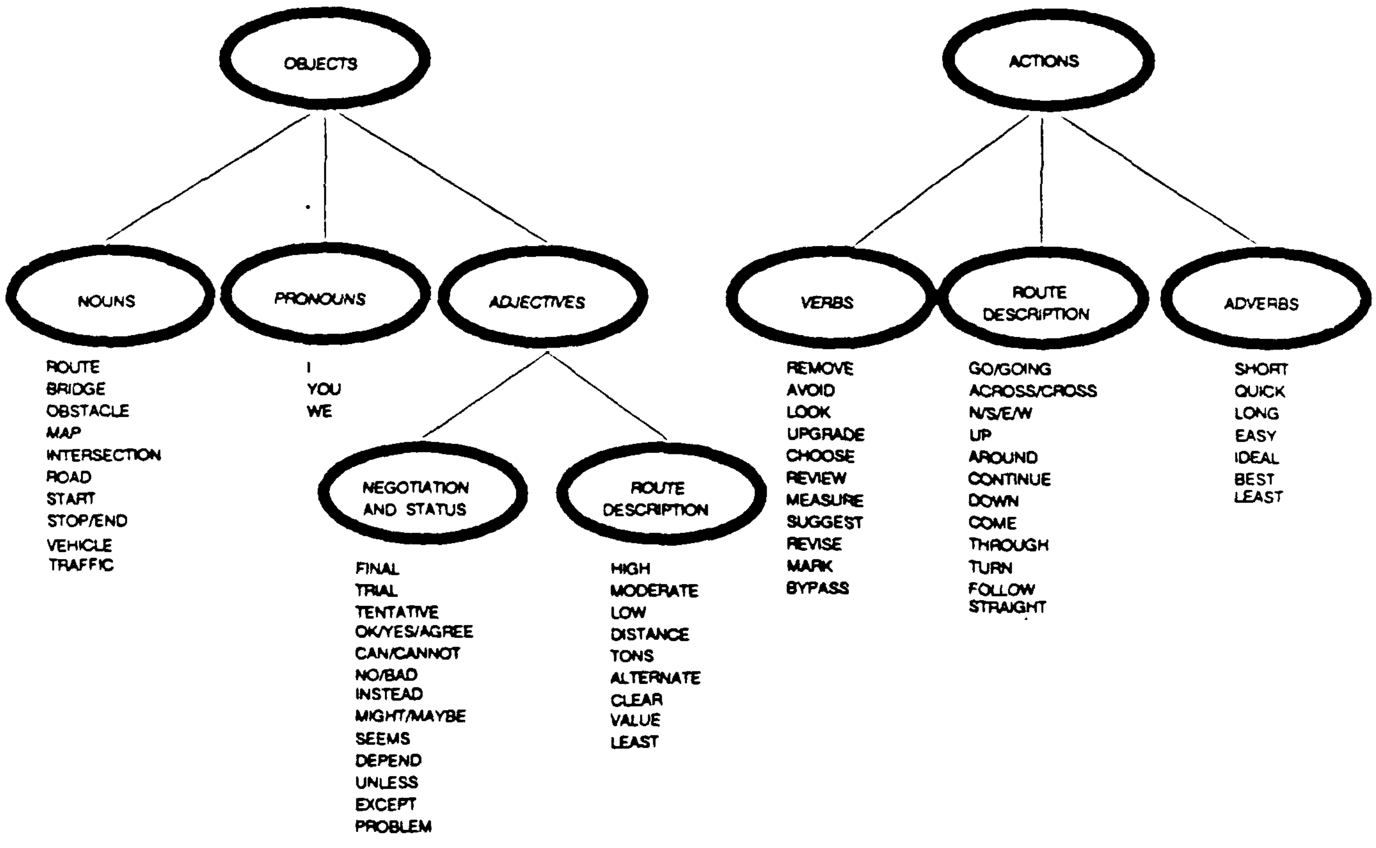


Figure 4-2. Example semantic map.
(from Linville, et al., 1989)

If the user has extensive experience with a specific data structure there will be a conflict if this structure is not included in the system software. For example, the user may have years of experience dealing with information stored in filing cabinets, with a specific organization, with folders with specific records, etc. In such a case, the data structure should be defined in the kind-of representation, or as a separate tabulation, and each item carefully defined.

Design relevance. The semantic analysis will provide an initial object and action vocabulary upon which dialogue design can be based. This analysis provides a basis for discussions among designers, users and programmers to achieve the semantic consistency and in turn yield low error, easy learning, and easy recall.

Syntactic Analysis

Purpose. The syntactic analysis represents the syntax, or grammar, of the language in a form which can be examined, and then used as a specification for software development.

Structure. Language representation used for computer languages can be adapted for dialogue design (cf., Phillips, Bashinski, Ammerman and Fligg, 1988, p. 853). Either transition diagrams (finite state diagrams), such as those used for specification of the PASCAL language, or Backus-Naur Form (BNF), such as those used for the FORTRAN language, may be modified for use. These show the states that can occur to the user, permissible courses of actions that can be taken from each state, and the transitions between states that result from a user's actions.

Extensions for the purpose of dialogue development, beyond the representations used for computer languages, have included:

- indicating whether a portion of the dialogue is generated by the user or the computer,
- indicating which of multiple users generates a command,
- indicating special display features such as blink, underline, etc., and
- echoing the last input when it is undecipherable by the computer, along with an error message.

A BNF and Transition Diagram representation for the same example is presented in Figure 4-3. Each construct is defined just once, insuring that competing definitions are not introduced into the dialogue design. The BNF definition is given in a top-down hierarchical fashion. First, a LOGON is defined as consisting of three parts: a welcoming message (LOGMSG), entry of a password (VALIDACCT), and a computer check of validity (CHECK). Then, each part is further defined. For example, VALIDACCT is defined as an account number followed by a password; the account number is defined as a message followed by an arbitrary number of numbers terminated by a carriage return. The Transition diagram presents the same information in graphical form.

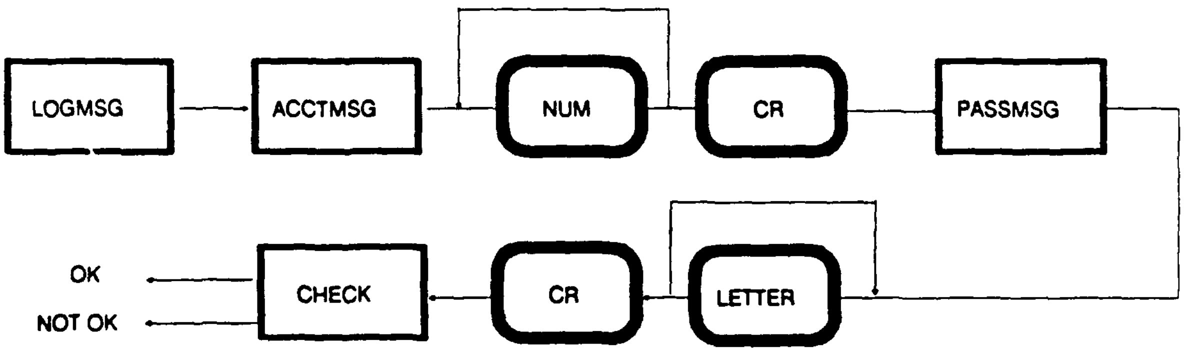
Design relevance. These analyses provide a method for examining the detailed implementation of a dialogue. Note that the BNF requires that each operation is identified only once. With care by the designer, syntactic consistency can be guaranteed.

SYMBOL	MEANING
:=:	AS DEFINED AS
< >	DEFINED CONSTRUCT
CAPS	DEFINED KEYWORD
lower case	BASIC SYMBOL OR DEFINITION
________	(UNDERLINE) REPEATABLE CONSTRUCT
|	CHOICE (OR)
{ }	MANDATORY CHOICE
[ ]	OPTIONAL CHOICE
,	SEPARATOR

(A) BACKUS-NAUR FORM DEFINITIONS

<LOGON>	:=:	<C:LOGMSG> <VALIDACCT> <CHECK>
<LOGMSG>	:=:	"WELCOME TO THE XYZ SYSTEM"
<VALIDACCT>	:=:	<ACCTNO> <PASSWORD>
<ACCTNO>	:=:	<C:ACCTNSG> <H:ACCTIN>
<PASSWORD>	:=:	<C:PASSMSG> <H:PASSIN>
<C:ACCTMSG>	:=:	"ENTER ACCONT NUMBER"
<C:PASSMSG>	:=:	"ENTER PASSWORD"
<H:ACCTIN>	:=:	<H:NUM> <CR>
<H:PASSIN>	:=:	<H:LETTER> <CR?
<H:NUM>	:=:	0|1|2|3|4|5|6|7|8|9
<H:LETTER>	:=:	A|B|C|...|Z
<CHECK>	:=:	<LOGIN.OK> | <LOGIN.NOT.OK>
<CR>	:=:	<CARRIAGE RETURN,RETURN KEY>

(B) BNF EXAMPLE, USER LOGIN PROCEDURE



(C) TRANSITION DIAGRAM, USER LOGIN PROCEDURE

Figure 4-3. Example syntactic representations.

A construct may be defined in terms of other constructs which are subsequently defined; this may make the representation difficult to read, but enhances its value for development of appropriate software. The analysis results in a specification which is directly usable by the programmer to produce executable code. A compiler can be produced for a specific BNF representation of a dialogue using available compiler-compiler software.

Lexical Analysis

Purpose. The lexical analysis deals with important issues associated with the vocabulary to be used in the user-computer dialogue. Specific analyses which may be performed are:

- selection of names identifiable and recallable by users,
- determination of suitable abbreviations, and
- computation of dialogue response time.

Structure. The selection of terms used in human-computer dialogue is important for achieving meaningful and low-error communication.

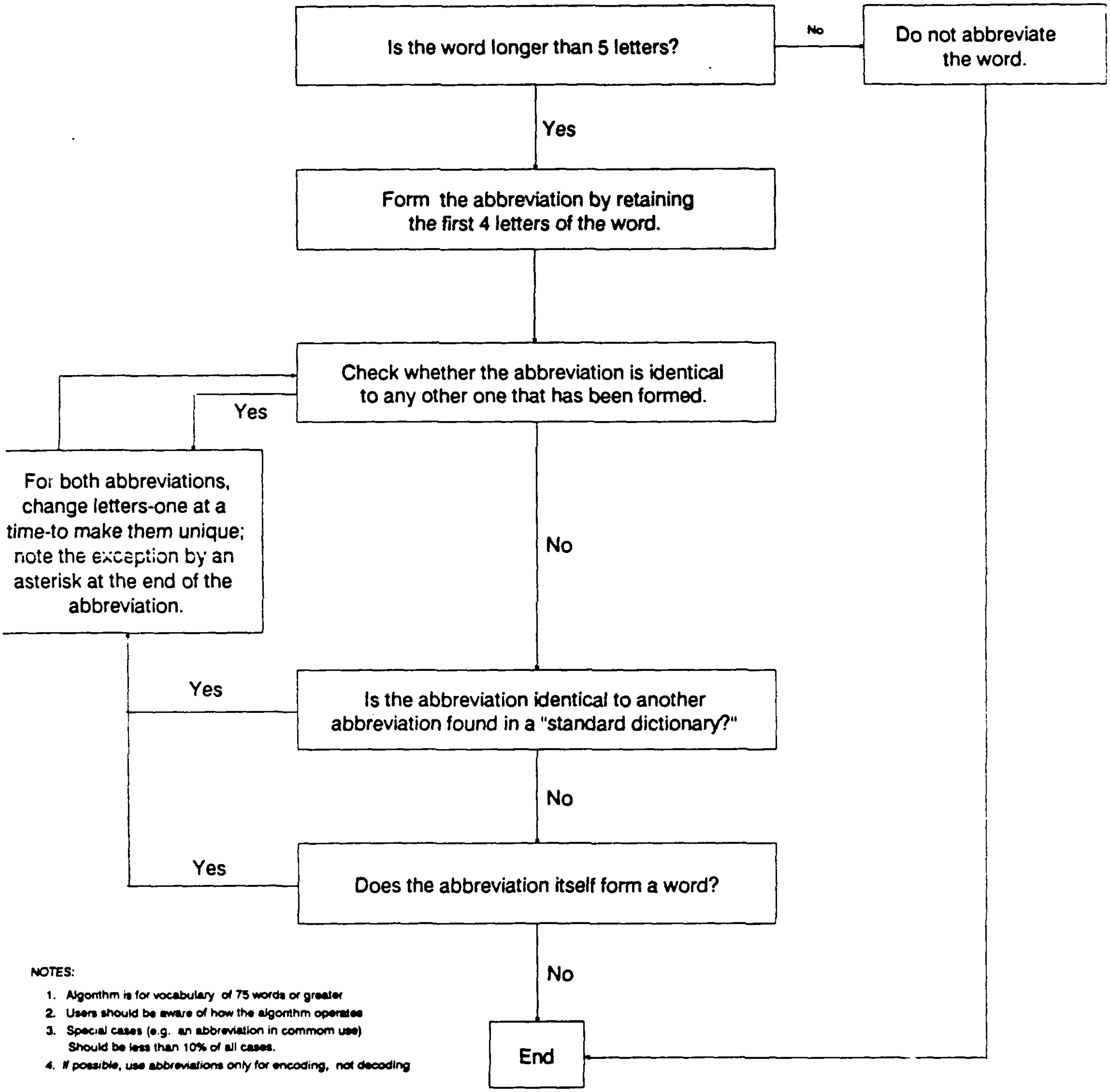
Ultimately selection of terms should be done by representative users. Subject matter experts and those reviewing relevant literature should develop lists of candidate synonyms, and representative users should then rate the candidates so that the best terms can be identified. When the class of users is homogeneous, as it may be for most command and staff users, the resulting ratings should yield a clear-cut selection of terms.

If novice or infrequent users are to use the system, both experts and novices can generate terms. The generation of candidate terms should follow experts' choices so the selection from the candidate list can be done by the novices (Bloom, 1987).

Abbreviating terms is often necessary or convenient; however, this should be done with care as abbreviating may obscure the identification of the term or cause confusion with other common terms. An algorithm for abbreviation (Moses and Potash, 1979), which has been subjected to empirical test, is presented in Figure 4-4.

In some situations (e.g., dialing a telephone, because this is done so frequently by so many, or responding to a threat, because of the need for fast response), even fractions of a second involved in dialogue can be important. If response time is important, models can be developed to predict task accomplishment time, and provide a basis for keystroke simplification as necessary.

For example, a Keystroke Model analysis (Card, Moran & Newell, 1983, p.259) provides a means to estimate the time required for an expert user to accomplish an interactive task with a computer system. A succinct set of rules are provided to estimate task execution time in terms of physical-motor operators (keystroking, pointing, homing and drawing), a mental operator and a system response operator. Execution time is simply the sum of the times spent executing the different operator types. To reduce the performance time of a task, one must eliminate operators from the method for doing the task. Application of the model to combat tasks will provide only an estimate, as the model does not handle disruptions.



NOTES:

1. Algorithm is for vocabulary of 75 words or greater
2. Users should be aware of how the algorithm operates
3. Special cases (e.g. an abbreviation in commom use) Should be less than 10% of all cases.
4. If possible, use abbreviations only for encoding, not decoding

Figure 4-4. Algorithm for abbreviation.
(from: Moses & Potash, 1979)

The lexical analysis will lead the designer to the need to make specific hardware and software assignments to the dialogue; that is, eventually one must decide *how* the language will be articulated. For example, one may wish to create a matrix of commands and interaction techniques to show any of the following which are applicable to each command (cf., Phillips, Bashinski, Ammerman & Fligg, 1988, p. 853):

- keyboard command/parameter,
- fixed-meaning function key,
- soft-meaning function key,
- cursor selection (via keys, mouse, trackball),
- keyboard field delimiter,
- keyboard "enter" key, and
- form-fill/menu "enter" option.

<u>Design relevance</u>. The lexical analysis provides a method for enhancing user interpretation and recall of the dialogue terms, and for ensuring acceptable dialogue response time.

<u>Screen Format Design</u>

<u>Purpose</u>. The purpose of screen format analysis and design is to identify the information to be displayed to the user as the computer portion of the dialogue.

<u>Structure</u>. The necessary screen design steps, as prescribed by Galitz (1989), are listed in Table 4-1. Most of the steps are common to all parts of a system development effort (for example, dialogue design); however, steps 4 and 5 are the ones to be emphasized.

After reviewing and identifying factors which affect the design in the first three steps of Table 4-1, the design process begins with a specification of important data elements to be included in each screen. Design worksheets should be prepared which include:

- title and name for each data element in a manner communicating clearly to the user,
- screen captions developed from the data element title,
- size of the field in character positions,
- frequency of occurrence of each data element (always required or optional), and
- logical relationships, or rules, for cross-checking with other data elements.

A transaction is a screen, or series of screens, which presents the output fulfilling a system requirement. These transactions are abstracted from design documents prepared during the system development phases, and data elements are identified that comprise each transaction.

The amount of display space required for each data element is computed (the sum of characters in the caption, the data field, and special attribute characters).

A very important step is to segment the data elements into logical groups, which can be based on sequence of use, frequency of use, function, and importance; but, the grouping must be consistent with the natural working habits of the system users.

The final layout is the result of judgments made based on available guidelines such as those presented in Galitz (1989).

Design relevance. This analysis results in the design of screen formats which constitute the computer-to-human portion of the dialogue.

Table 4-1

Screen Format Design Steps (Galitz, 1989)

I. Review screen design documentation and services
II. Identify system inputs and outputs
III. Identify unique user requirements
IV. Describe data elements
    A. Title/name
    B. Screen caption
    C. Size
    D. Required or optional status
    E. Logical relationship with other data elements
V. Develop transactions
    A. Summarize design requirements affecting screen design
    B. Specify data elements that will comprise a transaction
    C. Organize transaction data elements into sections
        1. Calculate data element lengths
        2. Apply grouping techniques and design considerations
        3. Specify necessary supplemental information
    D. Identify and layout screens
        1. Apply to each grouping (section) of data elements
        2. Conform to guidelines about screen line usage (about 3/4 of available width) and break screens at natural points
VI. Develop final paper screens
VII. Define computer screens
VIII. Test screens
IX. Implement screens
X. Evaluate screens

Section 5. Dialogue Design Guidelines

This section will present information to assist initial design of the human-computer interface for effective dialogue. First, general design principles will be presented and discussed. Next, specific dialogue types will be defined and guidelines presented for selection among dialogue types. Finally, dialogue interface recommendations will be summarized and illustrated with selected examples.

General Design Principles

General dialogue design principles are summarized in the following paragraphs. These are broad statements which may be applicable to any dialogue effort.

Make the dialogue directly relevant to the user. A notion of distance may be applied to the gulf between an individual's goals or knowledge and the level of description provided by the system. The designer should attempt to achieve directness, that is, a short distance. Match the level of description required by the interface language to the level at which the person thinks of the task. Also match meanings of expression to their physical input form.

For example, semantic directness may result by making the output show semantic concepts directly, such as using a WYSIWYG (what you see is what you get) display in a word processor. Physical-input directness may result by using a mouse to point to characters on the screen, as opposed to typing a row number and character string.

Reduce memory demands. Short-term memory is applied in recalling information soon after it is presented (perhaps a few seconds). Short-term memory is commonly limited to 7 plus or minus 2 items, but is limited even more for complex items. Long-term memory is used to recall information after longer periods of time and is affected by the organization of the information and associative networks which may be employed.

Memory demands can be reduced with dialogues which use recognition of items rather than recall, selection among 7 to 9 items or fewer, design of displays so that related information is all on one page, and providing on-line access to command syntax, abbreviations, codes and other information. Related issues are retention of training, and whether users will use a system frequently enough to be able to recall needed information.

Reduce error rate and significance. As much as is possible, a system should be designed so that serious error is not possible, and if an error is made, simple and understandable mechanisms should be provided for correction. Extensive dialogue should not have to be re-entered, but simple repair should be offered. Also, as much as is possible, any action should be reversible. For example, an UNDO command can be included. The user is thereby permitted to explore unfamiliar system features without high levels of anxiety.

Provide a standard and consistent dialogue. Sequences of action and terminology should be the same in similar circumstances. Formal specification of the language is a way to ensure consistency.

Consistency should be maintained, as much as is possible, with other computer operations that the user may encounter. For example, if the system is to be designed to be used with a commercially-available word processor, data base manager or a windows environment, the dialogue should be consistent with those systems.

Consider system response time. If the system response time is slow, the dialogue design should compromise by minimizing the number of system responses required. For example, if the time to re-write the display screen is long, or requires slow telecommunication, writing basic menus to the screen may be omitted as an option. Short response time is especially important to frequent users, and abbreviations, special commands, and macro facilities should be offered to them.

Keep the user informed and allow the user to be truly in charge. Every user action should result in some feedback information. Sequences of actions should be subdivided with specific feedback with regard to accomplishment, and to permit the user to revise or prepare for the next actions. The system design should not surprise the user or thwart the user in producing desired actions.

Alternative Types of Dialogue

Five types or styles of dialogue will be considered in the following paragraphs. Positive and negative features of each will be presented along with recommendations for use.

Menu selection. When a menu is presented, the users read the list, select the appropriate item, and indicate a selection in accordance with the syntax. Examples of several styles of menus are shown in Figure 5-1 and 5-2. Advantages and disadvantages of menu dialogue are listed in Table 5-1 Several of the disadvantages can be alleviated by implementing "macro" commands (which string individual selections into a single selectable macro) and using pop-up, tear-off, or walking menus to conserve display space.

Table 5-1

Advantages and Disadvantages of Menu Dialogue

Advantages	Disadvantages
1. Only recall is required, no memorization is necessary.	1. Menus can get large; there can be many items to select at a given time and there can be many levels in the hierarchy of lists. Many selections can be required to achieve a final selection known by the expert user from the start.
2. If terminology is meaningful in terms of the users task, and are distinct, little learning is necessary.	2. Menus require frequent revision of the display and therefore require a rapid display update rate.
3. A choice can be made with few keystrokes or moust activations.	3. Menus use display space, and may obscure parts of the display the user wishes to remain in view.
4. The user can be guided through a decision making sequence.	4. Menus are not useful for the entry of strings of alphanumeric characters (e.g., numbers, names).
5. Assuming only legal entries are ever presented on the menu, the user can not make a serious error. Other than making a syntax error in the selection process, the user can only select an inappropriate item and have to re-select the right one.	

Word Star: (Word Star is a registered trademark of MicroPro International Corporation)

< < < O P E N I N G   M E N U > > >

—Preliminary Commands—	—File Commands—	-System Commands-
L Change logged disk drive		R Run a program
F File directory now OFF	P PRINT a file	X EXIT to system
H Set help level		-WordStar Options-
—Commands to open a file—	E RENAME a file	T Run TelMerge
D Open a dicument file	O COPY a file	M Run MailMerge
N Open a non-document file	Y DELETE a file	S Run CorrectStar

(a) Main Wordstar menu

n   not editing

Use this command to create and alter program source files
and other non-documents. Word warp defaults off;
tabbing defaults to fixed (TAB chars in file; 8-col stops);
page breaks not shown; hi bit flags not used in file.
For normal word processing uses, use the "D" command instead.

A file name is 1-8 letters/digits, a period,
and an optional 0-3 character type.
File name may be preceded by disk drive letter A-D
and colon, otherwise current logged disk is used.

NAME OF FILE TO EDIT?

(b) Request for addditional information, Wordstar

Word Perfect: (Word Perfect is a registered trademark of Word Perfect Corporation)

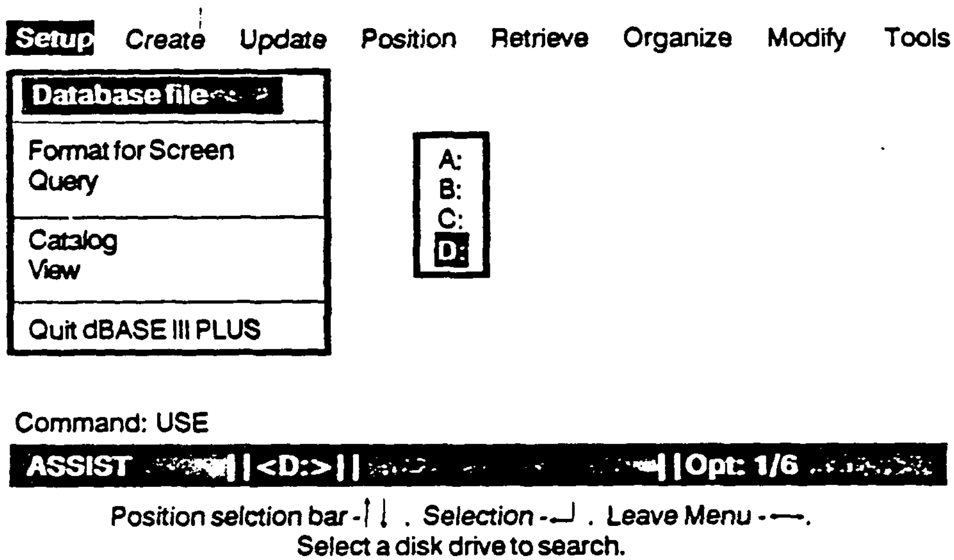
1 2 Tabs; 3 Margins; 4 Spacing; 5 Hyphenation; 6 Align Char; 0

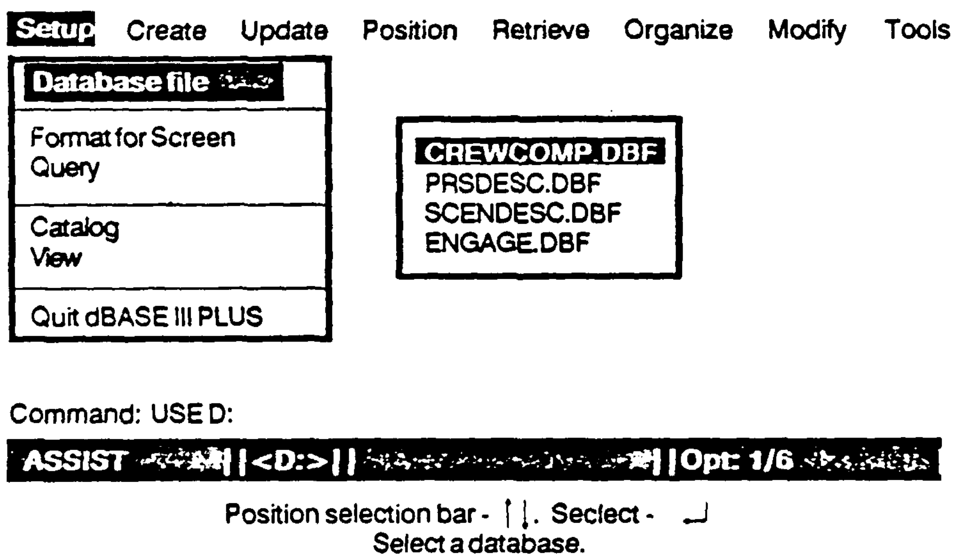
[Margin Set] 0 75 to Left = 5 Right = 65

(c) Horizontal, bottom -of-screen menus, Wordperfect

Figure 5-1. Example of menu dialogue used in word processors.

Intelligent Assistant Mode:





Command Mode:

USE D:CREWCOMP

(dBASE III + is a registered trademark of Ashton-Tate)

Figure 5-2. Example of pull-down menu and command language dialogue, database management (dBase III +).

Form fill-in. When data entry is required, a form with blank fields can be presented to the user. The user can move a cursor among the fields and enter data where desired. Form fill-in is a variation of a Question-and-Answer dialogue, in which the computer prompts the user for data. Form fill-in can be used for extensive data entry, or it can be used with menus to provide parameters (e.g., file names, system configuration) using a reduced form (sometimes called a dialogue box). An example of form-fill dialogue is shown in Figure 5-3; advantages and disadvantages are listed in Table 5-2.

Form Fill:

Table Modification Mode: **March Table Day1**

	Start	1st Stop	2nd Stop	3rd Stop	4th Stop
Ser	Time	Type Time/Time	Type Time/Time	Type Time/Time	Type Time/Time
1	0800	F 0930 /0945	/	/	/
2		/	/	/	/
3		/	/	/	/
4		/	/	/	/
5		/	/	/	/
8		/	/	/	/

Modify time for start of march for serial 1 on day 1

(D)elete, (M)odify, (L)eave Table

Figure 5-3. Example of form-fill dialogue, march table.

Table 5-2

Advantages and Disadvantages of Form-Fill Dialogue

Advantages	Disadvantages
1. User memory requirement can be minimized as computer can cue user with regard to content and format of the data to be entered.	1. The user should be able to enter data into data fields in any order, requiring some programming complexity and a modest level of training on how to move from field to field and how to edit the contents of fields. Under program control, the cursor should automatically jump between data fields and not position on labels or blank areas.
2. Help and advice can be offered, tailored to each data field, but increases programming complexity and display of help and advice must not obscure the data field.	2. Error checking must be provided for each data field and error messages must clearly indicate what form of correction is required.
3. Where a data field commonly contains the same value, the data field can default to this value provided the user can easily change it.	

Command language. Command language requires the user to enter a mnemonic code in response to a general system prompt. The codes trigger software which performs elemental functions, e.g. a set of tools, which the user may wish to combine in many configurations to suit specific requirements. An example of command language dialogue, contrasted with pull-down menu dialogue, is shown in Figure 5-2. Each darkened item in the menus indicates a menu selection which can be accomplished by one line of command language input. Advantages and disadvantages of command language are listed in Table 5-3.

Table 5-3

Advantages and Disadvantages of Command Language Dialogue

Advantages	Disadvantages
1. Allows the user flexibility in combining functions, and does not unnecessarily restrict the use of the system to preconceived tasks.	1. Error rates are typically high.
2. Seems to appeal to the "power" user, who may not wish to be delayed or distracted by computer prompting.	2. Training is necessary, and retention is likely to be poor.
	3. The diversity of possibilities and the difficulty of relating computer errors to user tasks makes it difficult to develop error messages or on-line help.

Natural language. Natural language dialogue enables the user to communicate with the computer in a fashion similar to person-person communication. In particular, a natural grammar is used and normal variations in form are allowed. However, in general, all computer natural language is restricted to some degree (and should be called Restricted Natural Language). Advantages and disadvantages of restricted natural language are listed in Table 5-4.

Table 5-4

Advantages and Disadvantages of Restricted Natural Language Dialogue

Advantages	Disadvantages
1. May be excellent where task domain is limited and training for grammar and syntax is infeasible.	1. User may have difficulty understanding the restrictions to natural language which are incorporated; results may be unpredictable.
	2. Implementation is very difficult and may require a large data base of language and task domain knowledge (additional difficulties are encountered if speech recognition is required).
	3. Dialogue may be slow, requiring many keystrokes and additional clarifying dialogue.

Direct manipulation. A direct manipulation interface involves a visual representation of a world of action familiar to a user, and allows the user to directly manipulate objects of interest within the visual representation. A simulation of a distillation process displayed graphically with fluid levels shown in tanks, and temperatures and volumes, is an example. Keyboard entries are generally replaced by pointing actions to select from a visible set of options (e.g. distillation process controls, see Figure 5-4). Advantages and disadvantages of direct manipulation are listed in Table 5-5.

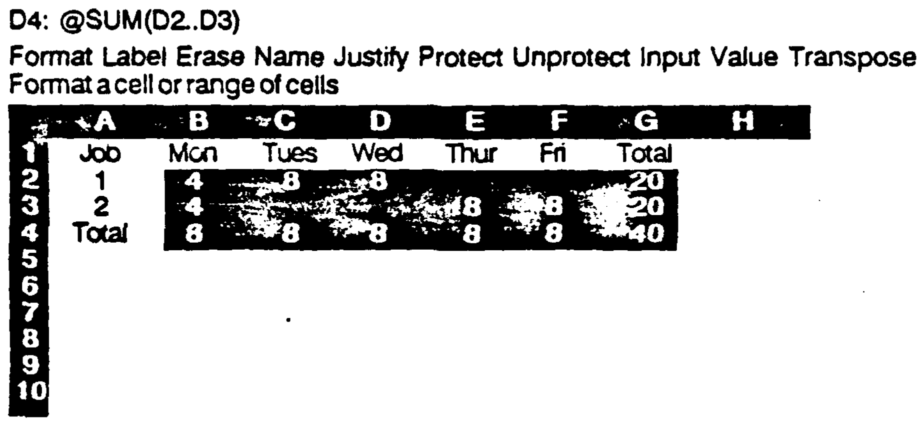
Table 5-5

Advantages and Disadvantages of Direct Manipulation Dialogue

Advantages	Disadvantages
1. Since there is an inherent directness, with a strong relationship to the user's mental model, this dialogue should be easy to learn and retain.	1. High-speed, high-resolution graphics and pointing devices are normally required. Programming is complicated and long development time is probable. This is a cost disadvantage.
2. Assuming that all actions available are legal and reversible, the user can explore the use of the system without serious error.	2. Augmentation with other types of dialogue may be necessary for commands and data entry, which do not lend themselves to a visual representation (e.g., the command structure within Lotus 1-2-3).
3. High user satisfaction is commonly reported.	3. Difficult to design and implement, (tries to match user's model rather than employing computer conventions).

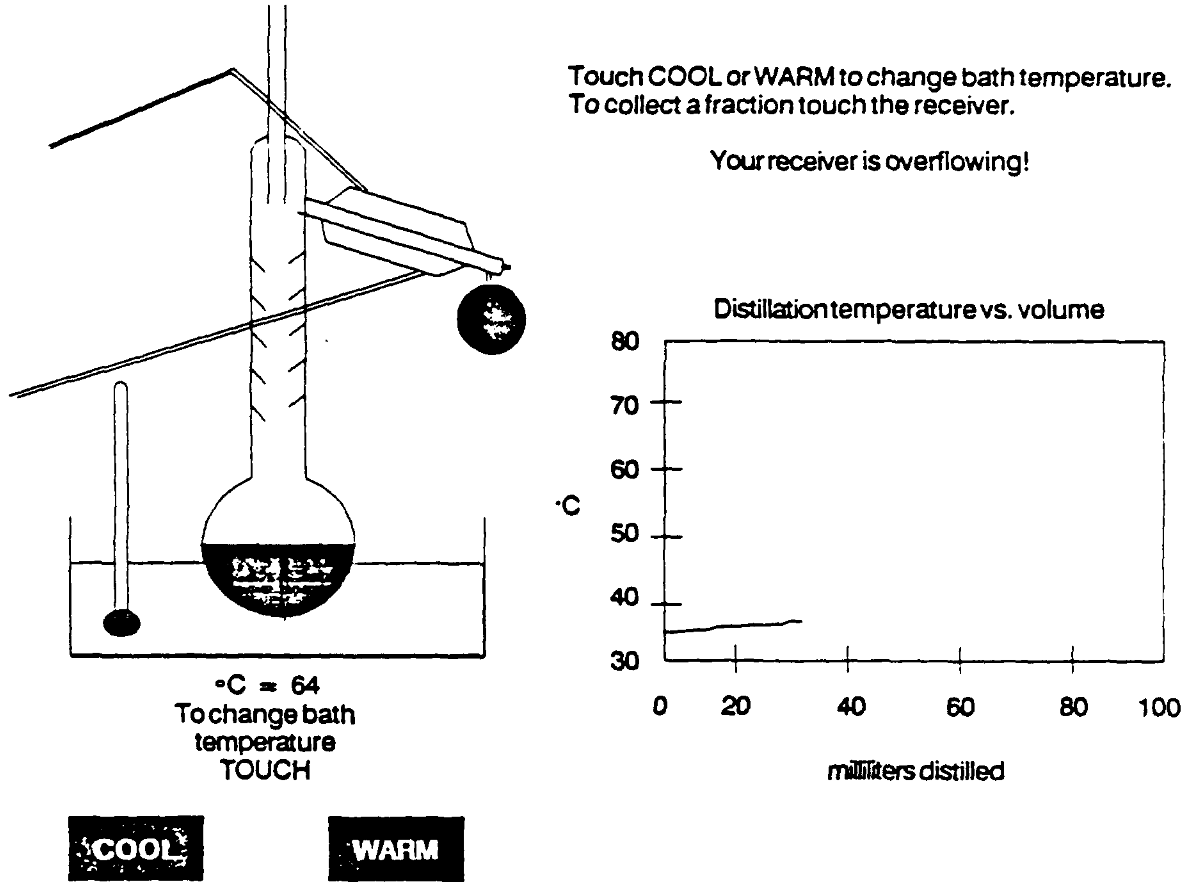
An abbreviated example of a direct manipulation dialogue is presented in Figure 5-5 and Figure 5-6. The user is presented with a "type" task organization (in this example a Mechanized Infantry Division). By use of a mouse controller and a selectable menu or command line, the user may view various units down to two levels below his assigned level, and may grab and drag a unit to another parent unit. All assigned units below the level being manipulated will also be dragged along with the parent. Once the task organization is satisfactorily modified, the user may generate the Task Organization paragraph for the Operations Order, Figure 5-7.

Lotus: (Lotus 1-2-3 is a registered trademark of Lotus Development Corporation)


D4: @SUM(D2..D3)
Format Label Erase Name Justify Protect Unprotect Input Value Transpose
Format a cell or range of cells

	A	B	C	D	E	F	G	H
1	Job	Mon	Tues	Wed	Thur	Fri	Total	
2	1	4	8	8			20	
3	2	4			8	8	20	
4	Total	8	8	8	8	8	40	
5								
6								
7								
8								
9								
10								

(a) Lotus 1-2-3



(b) Distillation Process Simulation

Figure 5-4. Examples of direct manipulation dialogue.
(from Shneiderman, 1983)

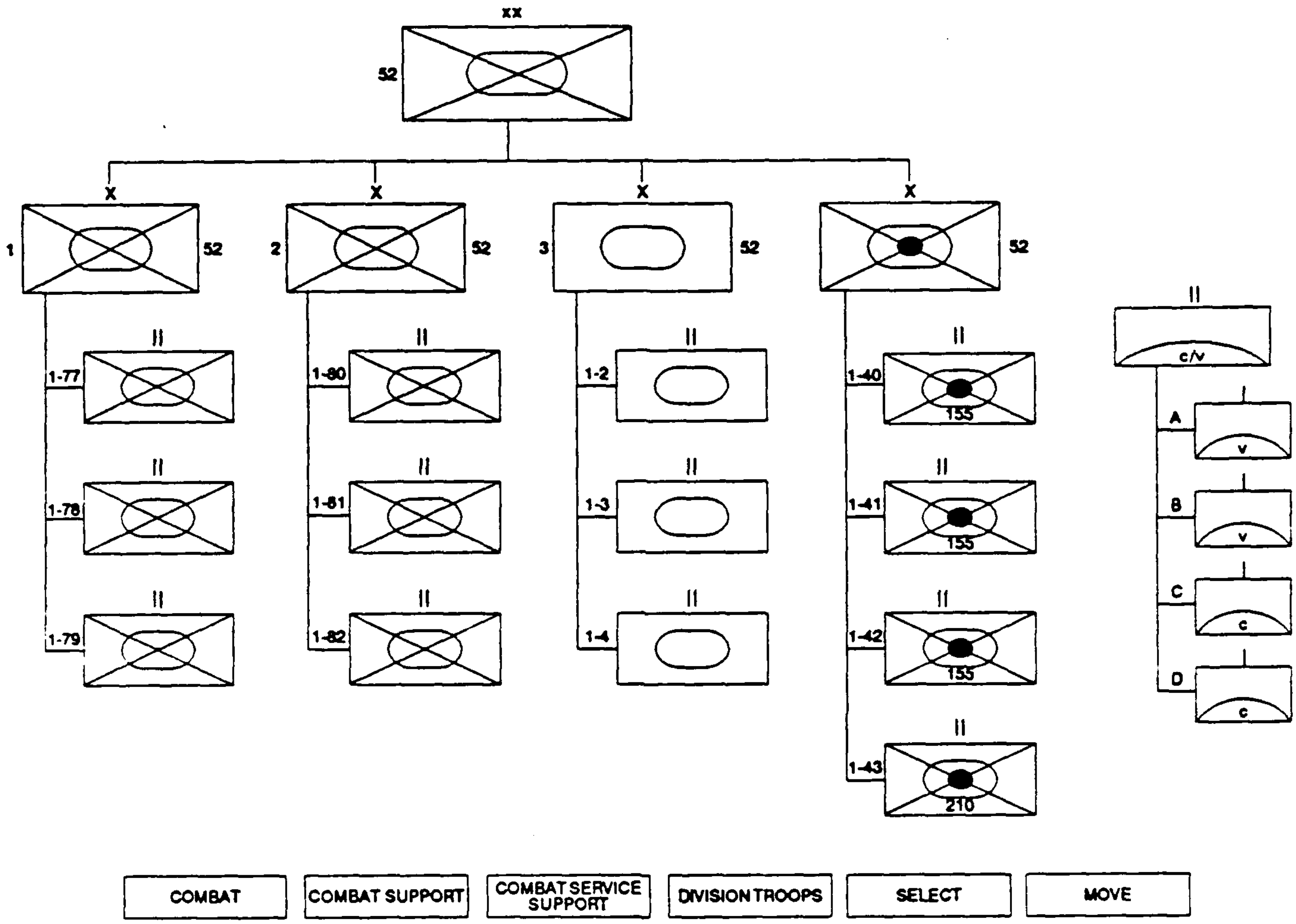


Figure 5-5. Task organization tool (partial example).

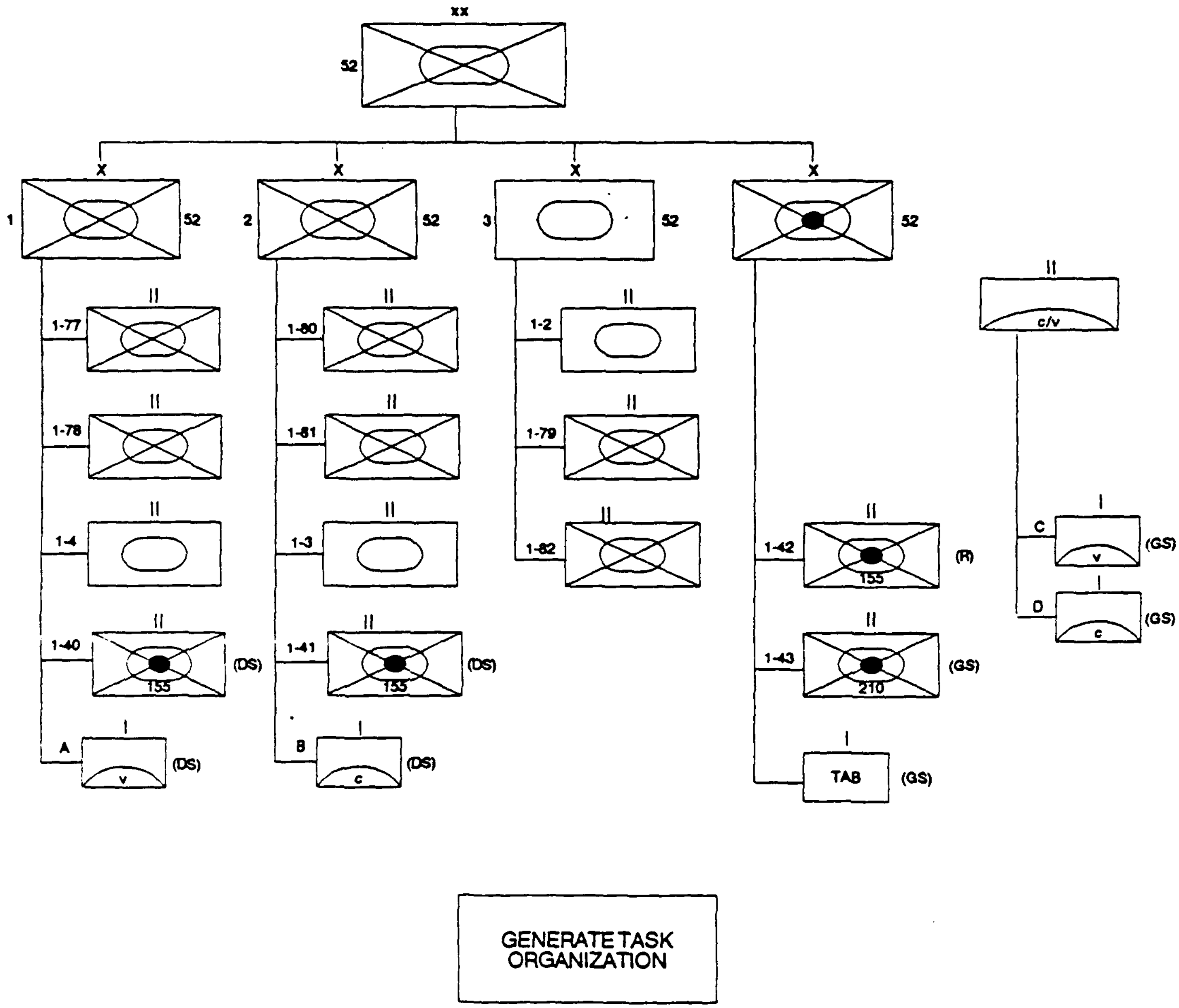


Figure 5-6. Complete task organization (partial example).

TASK ORGANIZATION: **(Partial listing only)**

1st Bde	3rd Bde
1-77 Mech	1-79 Mech
1-78 Mech	1-82 Mech
1-4 Armor	1-2 Armor
1-40 FA (DS)	
A/1-441 ADA (DS)	
2nd Bde	Div Arty
1-80 Mech	1-42 FA
1-81 Mech	1-43 FA
1-3 Armor	52d Tgt Acq Btry
1-41 FA (DS)	
B/1-441 ADA (DS)	
Div Trp	
1-441 ADA	

Figure 5-7. Task organization paragraph.

Combinations of Dialogue Types

Most computer operating systems are designed for command language dialogue, so this type of dialogue is usually easy to provide to the user in addition to other types. On the other hand, natural language dialogue is not easily provided unless a simple highly-restricted language is desired, or a software package is incorporated which includes a natural language interface. However, combinations of dialogue types are generally feasible and may be desirable for reasons to be discussed in the following paragraphs. In fact, most major software products will at least include command language, menu and form-fill in combination.

Accommodating a range of user types. Menu dialogue is commonly recommended for new and infrequent users, while command language is recommended for frequent experienced users. Many systems must provide for both types of users, and therefore include both menu and command language dialogue. The system can be designed to default to menu type with provision for the user to enter a code to switch to command language type. Note that expert users can be provided with fast execution with menu selection by providing a type-ahead feature, that is, by typing the first letter of each selection rapidly (sometimes referred to as the BLT type, since one would type BLT for selection of Bacon, Lettuce, Tomato options, also called "accelerators").

Accommodating a need for data entry. Menu selection type is of little use for entering data, although conceivably one could select from a menu including the alphabet, numbers, and special characters. Command language ordinarily provides for entry of parameters along with a mnemonic code (e.g., cp <Filename1> <Filename2> to copy the contents of one file to another). However, the equivalent with a menu selection would normally require use of a dialogue box, in which the user would enter the parameters that may be needed to execute the command. Similarly, direct manipulation dialogue may also require entry of data to augment the dialogue accomplished by pointing actions to the display.

Recommendations for Selection of Dialogue Type

Selection of the dialogue type depends on the characteristics of the system and of the expected users. For example, Sidorsky, Parrish, Gates & Munger, 1983, identify the following factors:

Number of commands. The total number of commands available to the user is a measure of the size of the dialogue vocabulary. Clear-cut recommendations are not possible based on command size alone. For large command sets (more than 150 commands), for example, menu selection is very difficult to use as a very deep hierarchy will result. On the other hand, large command sets pose a major training problem if command language is used. For medium command size, menu selection is generally recommended. For small command size (50 or less), size may be less important as a consideration, but menus may be selected to achieve some standardization. The key consideration is the grouping of commands and the user's recognition of the group. Familiar labels should be used to identify the groups.

Rate of use of commands. If an average command is used 5 or more times per day, command language is generally recommended, while menu selection is recommended if an average command is used less than 2 times per week.

Data transmission and display update rate. Dialogue types, such as menu select and direct manipulation, can significantly slow system response because of demands for re-writing the display and transmitting over a communication link to a host computer. When display and transmission rates are high (4800 baud or higher) the time delay is not a major factor.

Consistency. Many users have experience with programs on the same or other computers. If the dialogues are different for similar tasks, such users may experience interfering effects. Therefore, it is desirable to achieve standardization where possible, and in many cases it may be more important to standardize than to differ in the attempt to achieve small improvements. Through standardization, the collective experiences of users on other computer tasks can make him or her a sophisticated user for a new computer task.

Dialogue Technology in Popular Personal Computer Software

It is worthwhile for those interested in dialogue design to become familiar with a range of the current software available for personal computers. Such software and computer systems are readily available and may provide the subject for interesting dialogue evaluations. Those readers who may be developing small systems are especially urged to review the available software since their users are likely to be familiar with some personal computer software, and may have developed expectations based on this experience. For these reasons, a brief review of some popular personal computer software (for IBM-compatible and Zenith systems) will be conducted in the following paragraphs.

Word processors. Two of the most popular word processors are WordStar (WordStar is a registered trademark of WordStar® International Incorporated) and WordPerfect® (WordPerfect is a registered trademark of WordPerfect Corporated and is used under license with WP Corp and WP Corp reserves all rights therein); both are the result of long-term use and design iteration, but represent different approaches to dialogue design.

WordStar® originated in small 8-bit machines which did not have function keys or cursor-arrow keys. Consequently, all program selections are made using the control key in conjunction with ordinary keys used for typing. Therefore, all program control can be done without the user's hands being moved from the standard typing position. A hierarchy of menus is available to guide selection.

A basic menu (see Figure 5-1(a)) can permanently reside on the screen for inexperienced users, but can be eliminated for the experienced user. Further, users can type ahead of the display of menu selections, and if selections are entered rapidly enough the subsequent menus are not displayed. In this way, inexperienced users are shown each menu to aid selection, and experienced users do not see unnecessary menus cluttering the screen. As may be seen in Figure 5-1(b), menu selection may lead to requests for additional information to be typed using the keyboard.

WordPerfect® makes extensive use of function keys and attempts to leave the screen clear for the user's text. Four levels are provided for each function key, yielding a total of 40 selections by using each function key in combination with the shift, control, and alt keys. When necessary, a horizontal menu appears along the bottom of the screen (see Figure 5-1(c)). The inexperienced user may get help in using the function keys by touching the function key marked "help" and then touching any other key for which information is desired. The need for typing file names is minimized by use of a "list files" function key, which lists a directory of files for selection, and also allows features for browsing, searching for files with key words, deleting and other file maintenance functions.

Spreadsheets. Lotus 1-2-3® (Lotus is a registered trademark of Lotus Development Corporation) is a spreadsheet program and an example of a direct manipulation interface (see Figure 5-4(a)). Many functions are performed by moving the cursor to specific locations on the spreadsheet, or by indicating a specific block of cells of the spreadsheet. As the cursor is moved about the spreadsheet, information is presented on indicated cell in the upper-left corner of the screen. Other functions, like file selection or generation of graphs, are performed with menu selections.

Horizontal menus are used at the top of the screen and selections are made by typing the first letter of the selection or by moving the pointer. Either the next level of menu, or descriptive text, is presented just below the horizontal menu as an aid. When a file is to be selected for loading or storing, existing file names are presented to permit selection without typing.

Data bases. Data base management systems, and in particular languages for querying data bases, are a specialized area of human-computer dialogue (cf, Ehrenreich, 1981). Just two examples will be considered here: dBASE® (dBASE is a registered trademark of Ashton-Tate) and Q&A® (Q&A is a registered trademark of Symantec Corporation).

dBASE® has two basic modes: a command language mode, and a menu-driven assist mode (see Figure 5-2). In the assist mode, basic menu choices are shown in a horizontal bar near the top of the screen. When the cursor is placed on one of the basic menu choices, a pull-down menu appears and the desired alternative will be highlighted when the cursor is moved to that location, and the choice is made by pressing the return key. In this way, functions that deal with data base maintenance can be selected, such as creating, identifying, and making simple data retrieval. These are functions that might be assigned to clerical personnel. However, other functions involving complex retrieval and combining of data bases require use of the command language which is an extensive programming language.

Q&A® is data base management software like dBASE®, but, in addition to other interfaces, involves a natural language interface to the data base. It allows queries such as the following:

- What is the position of the JOUETT? [1]
- Her destination?
- How long would it take the KNOX to reach the PECOS?
- Reeves?
- What ships are within 1000 miles of Honolulu?
- List their readiness, reason, and casualty reports.

Q&A® allows an ordinary English language statement for retrieval, and does not require repeating all information with each request. It is a restricted natural language, but when a statement is not understood permits the user to train the system to respond properly.

Windows. "Windows" are areas of the computer screen which allow the user to have a simultaneous interface with multiple programs. The original window software used a desk top metaphor in which the output of each computer program can be viewed as a piece of paper or other object on a desk (with a number of pieces of paper overlapping or stacked on the desk surface). Other forms of window software use a "tiled" approach in which windows can overlap.

The windows can be treated as entities which, when not needed temporarily, can be replaced by a symbol, or icon, which can be placed at the edge of the screen. When needed again, the icon can be dragged to an appropriate place on the screen and expanded into a viewing window again. For control purposes, pull down menus are commonly used.

[1] Note: These are names of ships.

Section 6. Test and Evaluation

Whether design is conducted with nearly-continuous iterations (cf., Gould, 1988), or whether design is to be formally evaluated (cf., Williges & Hartson, 1986), there is a need for collecting information on the usefulness, usability, and user acceptance of the design. The methods and measures to produce such feedback will depend on whether it is early or late in the development process, the specific characteristics of the system and interface, the specific information needed (or lack of knowledge about the information needed), and the time and money available for testing.

In the following paragraphs, the range of methods which can be applied will be discussed first, followed by a discussion of the range of measurements which can be used to provide design feedback and evaluation.

Methods

Levels of simulation. Unless the goal is to evaluate a system for which a testable version exists, measurement must be collected using a simulation which represents the actual system. Studies based on such simulations may vary along a number of dimensions:

- Simple (inexpensive) -- complex (expensive): simulations may contain few or many of the represented system features; user-computer interaction studies require that the interface be included, but not necessarily in full detail.
- Abstract -- realistic: one may use paper and slide presentations or CRT displays, simplified models or working system elements.
- Non-interactive -- interactive: a simulation may provide to users a fixed sequence of events in accordance with a scenario, or permit interactive software for user-determined sequences of control.
- No quantitative performance measurement -- automated performance measurement: the provision for measurement may range from no measurement included in the simulation (then the designer must measure by some external means) to automated recording of timed events and calculation of desired measures.
- Informal methods -- formal methods: informal methods may involve a few users trying a system to find unforeseen problems and yielding a qualitative assessment, to data collection under controlled conditions with an experimental design and statistical analysis.

Testing early in the design process. A form of design review, early in the design process, uses a scenario which incorporates all system features to perform a typical sequence of tasks to accomplish a typical mission. The user then "walks through" this scenario, step by step, "using" the system features. At a very early stage, the presentation given to the user can be screen images on paper, possibly with a workstation mockup, so that the user can clearly visualize the system operation.

Specific design alternatives can be included at appropriate points in the walkthrough to derive user critique. At subsequent stages of system development, the simple simulations can be replaced with working elements, until finally, user response can be obtained with a highly-realistic representation of the system.

Tools for early-design study may take a variety of forms, including:

- paper prototypes,
- hypercard,
- slide shows, and
- "Wizard of Oz" techniques (people perform some machine functions).

Upon subsequent development, one may then include the following:

- stand-alone personal computer or workstation simulations,
- networked workstations, and
- instrumented workstations interfaced to real-world equipment.

Interactive rapid-prototype testing. Rapid prototyping is a term used to indicate methods which allow essential system features to be simulated and easily changed, and which permit dialogue design to converge to an acceptable product through successive iterations (cf., Harker, 1987; Hoyos, Gestalter, Strube & Zang, 1987; Myers, 1988). There are two basic types: incremental prototyping in which the final product is iteratively developed, and the throw-away approach in which the prototype is only used to clarify requirements and develop a system specification.

There are a number of advantages to be gained by this approach (Wassermand and Shewmake, 1985):

- it enables the user to evaluate the interface in practice and to suggest changes to the interface,
- it enables the developer to evaluate user performance with the interface and to modify it to minimize user errors and improve user satisfaction,
- it facilitates experimentation with a number of alternative interfaces and modification of interfaces,
- it gives the user a more immediate sense of the proposed system and thereby encourages users to think more carefully about the needed and desirable characteristics of the system, and
- it reduces the likelihood of project failure.

Hopefully, iterative design procedures lead to easy-to-use interfaces, reduce the expenses of software development and provide a useful tool in organizational development. As the user participates in the design through rapid prototyping, user satisfaction and motivation is increased, and communication may be improved among users, designers, and technicians.

The approach leads to reduced costs, for example (Weinschenk, 1989):

- reduced labor costs for development staff during initial development,
- reduced labor costs for development staff due to fewer changes after coding,
- reduced losses due to user rejection of the system, and
- reduced training requirements.

On the negative side, rapid prototyping leads to small samples of data, shoot-from-the-hip analyses and poor experimental methods. The users are frequently not typical of actual users, and the experience tends to be more of a snapshot rather than long-term involvement. Simulation for rapid prototyping tends to be difficult with complex systems. Prototyping problems may also include (Weinschenk, 1989): ignoring limitations and constraints, overselling expectations, and losing control.

Harker (1987) lists the following requirements for interface prototyping:

- realistic simulation of task scenario(s),
- representative sample of proposed user population,
- viable design options,
- systematically planned program of user tests,
- well-structured methods for data capture, and
- appropriate methods of data analysis and interpretation in a form which designers can use.

There is a growing collection of tools called User Interface Management Systems (UIMS) which contain a collection of interaction techniques from which an interface may be created. The collection of interaction techniques may include:

- physical input devices--mouse, keyboard, tablet, knobs,
- types of values--command, number, percent, location, name,
- dialogue types--menus, graphical sliders, on-screen light buttons,
- control--sequencing of events and interaction techniques, and
- analysis--helps to study and evaluate the user interface.

Examples of UIMSs are Peridot (Myers, 1988) and Sassafras (Hill,1987). Each of these is the result of a doctoral dissertation and support rapid prototyping. A recent development, SERPENT (Carnegie Mellon Univ., 1989), sponsored by the Department of Defense, is a media-independent UIMS which separates interface concerns from application concerns. Furthermore, SERPENT is not a throw-away prototype since the tool is also used for production. Given such tools, one may iteratively design the user interface long before the total human-computer system is developed.

Formal test and evaluation. Formal experimentation involves an experimental design, adequate quantities of users, quantitative measurement in a controlled setting, and a statistical analysis. Formal experimentation minimizes error and provides statistical comparisons of results (including an assessment of the potential error of interpretation). Consequently, formal experimentation should be included in test and evaluation whenever feasible. Such experimentation must be constrained even in the best of circumstances, since human-computer systems frequently involve the "curse of dimensionality" and a thorough study would include a large number of experimental comparisons. Except for the most extreme circumstances, the dedicated developer should use qualified scientists to conduct formal experimentation for at least the most important design considerations.

Measurement

Usability specification and measurement. Usability specifications provide precise, testable statements of performance goals for typical users carrying out tasks representative of their projected use of the system. The specifications are sufficiently detailed to show the behavioral prerequisites along with performance criterion. An example (Carroll and Rosson, 1985, p. 22) is:

"After successfully creating and printing a memo, 90% of a sample of secretaries with no word processing experience, using only the training materials provided, will be able to create a two-page report with an embedded table in 40 minutes."

Application of this orientation has been termed "usability engineering" (Bennett, Butler & Whiteside, 1989). Usability engineering has the following objectives:

- provide a quantitative operational definition of usability,
- set planned levels for usability attributes,
- analyze the impact of proposed design solutions,
- incorporate user-derived feedback into the evolving design, and
- iterate until planned levels are achieved.

The keys to the development of such specifications are the identification of pertinent test items and precise criteria. Some sources to consider in developing usability specifications are:

- current system performance data baseline,
- performance of similar systems,
- prediction of new requirements,
- user-derived information--design participation, interviews, surveys,
- selected design guidelines,
- pilot studies and rapid prototype tests, and
- previous specifications.

The usability engineering approach identifies for public discussion (Bennet, et al., 1989): What counts for success? How to measure success? How might we know in advance? The key attributes and measurement might be tabulated as follows:

ATTRIBUTE	MEASURED CONCEPT	WORST CASE VALUE	PLAN VALUE	BEST CASE VALUE
FUNCTION COST SCHEDULE USABILITY				

The attribute USABILITY can be further divided into:

- Learnability -- mastery of basic operations, ease of learning, rate of learning, transfer of learning (e.g., time to learn major functions, retention of commands over time),
- Throughput -- performance by skilled workers, productivity, power use (e.g., speed of task performance, rate of errors), and
- Satisfaction -- quality of user experience, user attitude (e.g., subjective responses).

If a working computer simulation is developed for design test, it is important to include a general purpose data collection program for unobtrusively storing user performance variables for later analysis. As part of this program, user actions are intercepted and stored along with a time stamp (with msec tolerances if possible); additionally an experimenter station is desirable to record experimenter codes and comments (with time) to supplement the keystroke-level data collection.

Some measurements to consider for quantitative evaluation include (Shaw and McCauley, 1985):

- time to complete a training program,
- time to achieve a performance criterion,
- observed difficulty in learning a product,
- user comments, suggestions, and preferences,
- time to perform selected tasks,
- success in task completion,
- frequency of use of commands or language features,
- time spent in locating information in documentation,

- inability to find information in documentation,
- frequency that each error message is encountered,
- frequency of use of on-line help, and
- use of special assistance.

Consideration should be given to retaining a recording of user activity as a long-term, or permanent, part of the system. Long-term analyses of the use of system features, and analyses for trouble-shooting, then can be accomplished after use of the system has stabilized.

Section 7. References

Badre, A. and Shneiderman, B. (1984). Directions in human/computer interaction. Norwood, NJ: Ablex.

Baker, C., Eike, D. R., Malone, T. B., and Peterson, L. (1988). Update of DoD-HDBK-761: Human engineering guidelines for management information systems. Proceedings of the Human Factors Society, 32nd Annual Meeting (pp. 335-339). Santa Monica, CA: Human Factors Society.

Bennett, J., Butler, K., Whiteside, J. (1989). Usability engineering. Tutorial presented at CHI'89 Conference. Austin, TX: ACM, Special Interest Group on Computer and Human Interaction.

Card, S. K., Moran, T. P. and Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum.

Carnegie Mellon University (1989). SERPENT, a media independent user interface management system. Pittsburgh, PA: CMU, Software Engineering Institute.

Carroll, J. M. and Rosson, M. B. (1985). Usability specifications as a tool in iterative development. In Hartson, H. R. (Ed.), Advances in Human-Computer Interaction. Norwood, NJ: Ablex.

Combined Arms Operations Research Activity (CAORA) (1985). ACN 70876 G3 analysis: Volume I: Main Report. Volume II: Appendices. (CAORA/TR-13/85). Fort Leavenworth, KS: Author.

Crumley, L. (1989). Review of research and methodologies relevant to Army Command and Control performance measurement (Technical Report 825). Ft. Leavenworth, KS: U.S. Army Research Institute Field Unit. AD A211 247

Department of Defense. (1985). Military Handbook: Human Engineering Guidelines for Management Information Systems (DOD-HDBK-761). Washington, DC: Author.

Ehrenreich, S. L. (1981). Query languages: design recommendations derived from the human factors literature. Human Factors, 23, 709-725.

Galitz, W. O. (1989). Handbook of screen format design (3rd Edition). Wellesley, MA: QED Information Sciences, Inc.

Gould, J. D. (1988). Designing for usability: the next iteration is to reduce organizational barriers. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp. 1-9). Santa Monica, CA: Human Factors Society.

Gould, J. D. and Lewis, C. (1985). Designing for usability: Key principles and what designers think. In Shackel, B. (Ed.), INTERACT '84: First conference on human-computer interaction. Amsterdam: North-Holland.

Halpin, S. M. (1984). A proposal for an intelligent interface in man-machine systems. Proceedings of the 23rd IEEE Conference on Decision and Control, 592-595.

Harker, S. (1987). Rapid prototyping as a tool for user centered design. In Salvendy, G. (Ed). Cognitive engineering in the design of human-computer interaction and expert systems. Proceedings of the Second International Conference on Human-Computer Interaction, Honolulu, HI. Amsterdam: Elsevier.

Hartson, H. R., Roach, J. W., Callan, J. E. III, Nickson, M. M., Pittman, J. A. and Yunten, T. (1986). Dialogue management as part of software engineering methodology. In Ehrich, R. E. and Williges, R. C. (Eds.), Human-Computer Dialogue Design. New York: Elsevier.

Hayes, P. J. (1985). Executable interface definitions using form-based interface abstractions. In Hartson, H. R. (Ed.),Advances in Human-Computer Interaction. Norwood, NJ: Ablex

Hill, R. D. (1987). Supporting concurrency, communication and synchronization in human-computer interaction. Toronto, Canada: University of Toronto. (PhD Thesis, M5S1A4)

Hoyos, C. G., Gestalter, H., Strube, V. and Zang, G. (1987). Software design with the rapid prototyping approach: A survey and some empirical results. In Salvendy, G. (Ed). Proceedings of the Second International Conference on Human-Computer Interaction, Honolulu, HI.

Hurd, J. C. (1985). Standardizing the software user interface. Human Factors Society Bulletin, 28(9), 3-4.

Jacob, R. J. K. (1986). An executable specification technique for describing human-computer interaction. In Hartson, H. R. (Ed.)Advances in Human-Computer Interaction. Norwood, NJ: Ablex Publishing Corp.

Kammersgaard, J. (1988). Four different perspectives on human-computer interaction. International Journal of Man-Machine Studies, 28, 343-362.

Kincade, R. G. and Anderson, J. (Eds.). (1984). Human factors guide for nuclear power plant control room development. Palo Alto, CA: Electric Power Research Institute.

Linville, J. M., Obermayer, A. H., Obermayer, R. W., and Fallesen, J. J. (1989). A method for the development of a graphic communication language for computer-mediated distributed command and control systems. Thousand Oaks, CA: VRC Corporation.

Mancini, G., Woods, D. D. and Hollnagel, E. (Eds.) (1987). Special issue: Cognitive engineering in dynamic worlds. International Journal of Man-Machine Studies, 27(5&6), 459-751.

Mark, W. (1986). Knowledge-based interface design. In Norman, D. A. and Draper, S. W. (Eds.). User Centered System Design. Hillsdale, NJ: Lawrence Erlbaum.

Martin, J. (1973). Design of man-computer dialogues. Englewood Cliffs, NJ: Prentice-Hall.

Meister, D. and Sullivan, D. J. (1967). A further study of the use of human factors information by designers. Canoga Park, CA: The Bunker-Ramo Corp.

Meister, D., Sullivan, D. J. and Finley, D. L. (1969). The effect of amount and timing of human resources data on subsystem design. Canoga Park, CA: The Bunker-Ramo Corp.

Miller, R. B. (1956). A suggested guide to position-task description (ASPRL-TM-56-6). Lowry Air Force Base, CO: Air Force Personnel and Training Research Center.

Monk, A. (1985). Fundamentals of human-computer interaction. San Diego, CA: Academic Press.

Moses, F. L. and Potash, L. M. (1979). Assessment of abbreviation methods for automated tactical systems (ARI Technical Report 398). AD A077 840

Myers, B. A. (1988). Creating user interfaces by demonstration. Boston: Academic Press.

Nickerson, R. S. (1981). Why interactive computer systems are sometimes not used by people who might benefit from them. International Journal of Man-Machine Studies, 15(4), 469-483.

Norman, D. A. and Draper, S. W. (1986). User centered system design: new perspectives on human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum.

Phillips, M. D., Bashinski, H. S., Ammerman, H. L. and Fligg, C. M. (1988). A task analytic approach to dialogue design. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: Elsevier.

Rasmussen, J. (1986). Information procession and human-machine interaction: An approach to cognitive engineering. New York, NY: North-Holland.

Reidel, S. (1988). User acceptance and field implementation of decision support systems (ARI Research Report 1477). AD A200 412

Rouse, W. B. and Boff, K. R. (1987). System design: behavioral perspectives on designers, tools, and organization. New York: North-Holland.

Sage, A. P. (Ed.) (1987). System design for human interaction. New York: IEEE Press.

Shaw, B. E. and McCauley, M. E. (1985). Person computer dialogue: A human engineering data base supplement (AFAMRL-TR-85-013). Wright-Patterson AFB, OH: Air Force Aerospace Medical Research Laboratory.

APPENDIX A
SUGGESTED READING

Card, S. K., Moran, T. P. and Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Assoc.

Hartson, H. R. (Ed). Advances in human-computer interaction. Volume 1. Norwood, NJ: Ablex.

Helander, M. (Ed.) (1988). Handbook of human-computer interaction. Amsterdam: North-Holland.

Monk, A. (1985). Fundamentals of human-computer interaction. San Diego, CA: Academic Press.

Norman, D. A. and Draper, S. W. (1986). User centered system design: new perspectives on human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Assoc.

Sage, A. P. (Ed.) (1987). System design for human interaction. New York, NY: IEEE Press.

Shneiderman, B. (1986). Designing the User Interface: Strategies for effective human-computer interaction. Menlo Park, CA: Addison-Wesley.

Ehrich, R. W. and Williges, R. C. (Eds) (1986). Human-computer dialogue design. New York: Elsevier.

Williges, B. H. and Williges, R. C. (1984). Dialogue design considerations for interactive computer systems. In Muckler, F. A. (Ed.) Human Factors Review. Santa Monica, CA: Human Factors Society.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. IEEE Computer, 16(8), 57-69.

Shneiderman, B. (1986). Designing the User Interface: Strategies for effective human-computer interaction. Menlo Park, CA: Addison-Wesley.

Sidorsky, R. C., Parrish, R. N., Gates, J. L and Munger, S. J. (1983). Design guidelines for user transactions with battlefield automated systems: Prototype for a handbook. ARI Research Product 84-08. AD A153 231

Simes, D. K and Sirsky, P. A (1985). Human Factors: An exploration of the psychology of human-computer dialogues. In Hartson, H. R. (Ed.) Advances in human-computer interaction (Vol. 1). Norwood, NJ: Ablex.

Smith, S. L and Mosier, J. N. (1984). Design guidelines for user-system interface software (ESD-TR-84-190). Hanscom AFB, MA: Electronic Systems Division, Air Force Systems Command. AD A154 907

Softech, Inc. (1978). Integrated computer-aided manufacturing (ICAM) Task 1 Final Report (AFML-TR-148). Wright-Patterson AFB, OH: Air Force Materials Laboratory.

Solick, R. E. and Lussier, J. W. (1988). Design of battle simulations for command and staff training (Technical Report 788). Ft. Leavenworth, KS: U.S. Army Research Institute Field Unit. AD A196 655

Wasserman, A. I. and Shewmake, D. T. (1985). The role of prototypes in the user software engineering (USE) methodology. In Hartson, H. R. (Ed.), Advances in human-computer interaction, 1, (pp. 191-209). Norwood, NJ: Ablex.

Weinschenk, S. (1989). Rapid prototyping: human factors techniques and tools. Tutorial presented at CHI'89 Conference. Austin, TX: ACM, Special Interest Group on Computer and Human Interaction.

Williges, R. C. and Hartson, H. R. (1986). Human-computer dialogue design and research issues. In Ehrich, R. W. and Williges, R. C. (Eds.), Human-computer dialogue design. New York: Elsevier.

Williges, B. H. and Williges, R. C. (1984). Dialogue design considerations for interactive computer systems. In Muckler, F. A (Ed.), Human Factors Review. Santa Monica, CA: Human Factors Society.

APPENDIX B
SELECTED BIBLIOGRAPHY

This section contains a bibliography corresponding in scope to the main body of this document. Since an excellent review of the literature was published in 1984 (Williges and Williges, 1984), this bibliography includes only reports not included in that review (i.e., approximately the last five years). Only reports which focus on dialogue issues are included, and reports are excluded which deal primarily with display, input/output devices, feedback and error management, security, documentation and training. The index below provides a key to the citation numbers in the bibliography which follows.

Index to Bibliography (keyword, citation number)

Index to Selected Bibliography (cont'd.)

1 Akscyn, R.M., Yoder, E., McCracken, D. (1988). The data model is the heart of interface design. CHI'88 Conference Proceedings (pp.115-120). Washington, DC: ACM/SIGCHI.

2 Alavi, M. (1984). An assessment of the prototyping approach to information systems development. Communications of the ACM, 27(6), 556-563.

3 Allwood, C.M. (1986). Novices on the computer: a review of the literature. International Journal of Man-Machine Studies, 25(6), 633-658.

4 Anacker, P. (1987). Intelligent interfaces. PC AI, 1(3), 49-53.

5 Anderson, N.S. and Reitman-Olson, J. (Eds) (1985). Methods fordesigning software to fit human needs and capabilities. Washington, DC: The National Academy of Sciences/National Research Council Committee on Human Factors.

6 Arthur, J.D. (1986). A descriptive/prescriptive model for menu-based interation. International Journal of Man-Machine Studies, 25(1), 19-33.

7 Backs, R.W., Walrath, L.C. and Hancock, G.A. (1987). Comparison of horizontal and vertical menu formats. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.715-717). New York, NY: Human Factors Society.

8 Badre, A. and Shneiderman, B. (1984). Directions in human/computer interaction. Norwood, NJ: Ablex.

9 Baker, C.C.,Eike, D.R., Malone, T.B. and Peterson, L.A. (1988). Update of DoD-HDBK-761: "Human engineering guidelines for management information systems." In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.335-339). Anaheim, CA: Human Factors Society.

10 Barker, P.G., Najah, M. and Manji, K.A. (1987). Pictorial communication with computers. International Journal of Man-Machine Studies, 27, 315-336.

11 Barnard, P.J. and Grudin, J. (1988). Command names. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

12 Barnhard, P., MacLean, A. and Wilson, M. (1988). Navigating integrated facilities: initiating and terminating interaction sequences. CHI'88 Conference Proceedings (pp.121-126). Washington, DC: ACM/SIGCHI.

13 Baroff, J., Simon, R., Gilman, F. and Shneiderman, B. (1987). Direct manipulation interfaces for expert systems. In J. Hendler (Ed.), Expert systems: the user interface. Norwood, NJ: Ablex.

14 Bauer, D.W. and Eddy, J.K. (1986). The representation of command language syntax. Human Factors, 28(1), 1-10.

15 Bayerl, J.P., Millen, D.R. and Lewis, S.H. (1988). Consistent layout of function keys and screen labels speeds user responses. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.344-346). Anaheim, CA: Human Factors Society.

16 Benbasat, I. and Wand, Y. (1984). Command abbreviation behavior in human-computer interaction. Communications of the ACM, 27(4), 376-383.

17 Benbasat, I. and Wand, Y. (1984). A structured approach to designing human-computer dialogues. International Journal of Man-Machine Studies, 21(2), 105-126.

18 Bloom, C.P. (1987). Procedures for obtaining and testing user-selected terminologies. Human-Computer Interaction, 3(2), 155-177.

19 Borgman, C.L. (1986). The user's mental model of an information retrieval system: an experiment on a prototype online catalog. International Journal of Man-Machine Studies, 24(1), 47-64.

20 Britts, S. (1987). Dialog management in interactive systems: a comparative survey. SIGCHI bulletin, 18(3), 30-42.

21 Brown, D.P., Sharratt, B. and Norman, M. (1986). The formal specification of adaptive user interfaces using command language grammar. In CHI'86 Conference Proceedings (pp.256-260). Boston, MA: ACM/SIGCHI.

22 Brown, C.M. (1988). Human-computer interface design guidelines. Norwood, NJ: Ablex.

23 Buie, E.A. (1986). Bibliography: individual differences and computer-human iteraction. SIGCHI bulletin, 17(3), 47-49.

24 Butler, K.A. (1985). Connecting theory and practice: a case study of achieving usability goals. In CHI'85 Conference Proceedings (pp.85-88). San Francisco, CA: ACM/SIGCHI.

25 Callahan, J., Hopkins, D. and Shneiderman, B. (1988). An empirical comparison of pie vs. linear menus. CHI'88 Conference Proceedings (pp.95-100). Washington, DC: ACM/SIGCHI.

26 Card, S.K., Moran, T.P. and Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Assoc.

27 Card, S.K. and Henderson, D.A., Jr. (1987). A multiple, virtual-workspace interface to support user task switching. In CHI + GI 1987 Conference Proceedings (pp.61-68). Toronto, Canada: ACM/SIGCHI and ACM/SIGGRAPH.

28 Carey, T. (1988). ACM self-assessment procedure on human-computer interaction. SIGCHI bulletin, 20(1), 26-30.

29 Carroll, J.M., Mack, R.L. and Kellogg, W.A. (1988). Interface metaphors and user interface design. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

30 Carroll, J.M. and Olson, J.R. (1988). Mental models in human-computer interaction. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

31 Carroll, J. M. and Rosson, M. B. (1985). Usability specifications as a tool in iterative development. In Hartson, H. R. Advances in Human-Computer Interaction. Norwood, NJ: Ablex Publishing Corp.

32 Carter, J.A. Jr. (1986). A taxonomy of user-oriented functions. International Journal of Man-Machine Studies, 24(3), 195-292.

33 Chechile, R.A., Fleischman, R.N. and Sadoski, D.M. (1986). The effects of syntactic complexity on the human-computer interaction. Human Factors, 28(1), 11-22.

34 Cherry, J.M. (1986). An experimental evaluation of prefix and postfix notation in command language syntax. International Journal of Man-Machine Studies, 24(4), 365-374.

35 Chignell, M.H. and Hancock, P.A. (1988). Intelligent interface design. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

36 Chin, J., Diehl, V., and Norman, K. (1988). Development of an instrument measuring user satisfaction of the human-computer interface. CHI'88 Conference Proceedings. Washington, DC: ACM/SIGCHI.

37 Clarke, A.A. (1986). A three-level human-computer interface model. International Journal of Man-Machine Studies, 24(6), 503-518.

38 Cochran, D.R. and Stocker, F.R. (1985). RIPL: an environment for rapid prototyping with intelligent support. SIGCHI bulletin, 17(2), 29-36.

39 Coleman, W.D., Williges, R.C. and Wixon, D.R. (1985). Collecting detailed user evaluations of software interfaces. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.240-244). Baltimore, MD: Human Factors Society.

40 Davies, S.E., Bury, K.F. and Darnell, M.J. (1985). An experimental comparison of a windowed vs. a non-windowed operating system environment. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.250-254). Baltimore, MD: Human Factors Society..

41 Desain, P. (1986). Direct manipulation and the design of user interfaces (REPT-87-FU-01). Nijmegen (Netherlands): Katholieke Univ. (PB88-126354/XAB)

42 Dumais, S.T. and Wright, A.L. (1986). Reference by name vs. location in a computer filing system. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.824-828). Dayton, O: Human Factors Society.

43 Edwards, J.L. and Mason, J.A. (1988). Toward intelligent dialogue with ISIS. International Journal of Man-Machine Studies, 28(2&3), 309-342.

44 Egido, C. and Patterson, J. (1988). Pictures and category levels as navigational aids for catalog browsing. CHI'88 Conference Proceedings (pp.127-132). Washington, DC: ACM/SIGCHI.

45 Ehrenreich, S.L. (1985). Computer abbreviations: evidence and synthesis. Human Factors, 27(2), 143-157.

46 Eklundh, K.S., Marmolin, H. and Hedin, C. (1985). Experimental evaluation of dialogue types for data entry. International Journal of Man-Machine Studies, 22(6), 651-661.

47 Elkerton, J. (1985). Modeling expert computer users to aid inexperienced computer users. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.851-855). Baltimore, MD: Human Factors Society.

48 Elkerton, J. (1986). A behavioral evaluation of command-selection aids for inexperienced computer users (Doctoral thesis). Blacksburg, VA: Virginia Polytechnic Institute and State Univ.

49 Elm, W.C. and Woods, D.D. (1985). Getting lost: a case study in interface design. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.927-931). Baltimore, MD: Human Factors Society.

50 Fairhurst, M.C., Bonaventura, M. and Stephanidis, C. (1987). Human-computer interaction in the provision of an interpersonal communication mechanism for the nonvocal. International Journal of Man-Machine Studies, 27(4), 401-412.

51 Farooq, M.U. and Dominick, W.D (1988). A survey of formal tools and models for developing user interfaces. International Journal of Man-Machine Studies, 29(5), 479-496.

52 Fink, P.K., Sigmon, A.H. and Biermann, A.W. (1985). Computer control via natural language. IEEE Transactions on Systems, Man & Cybernetics, SMC-15(1), 54-68.

53 Fisher, D.L., Coury, B.G. and Tengs, T.O. (1985). Optimizing the set of highlighted options on video display terminal menus. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.650-654). Baltimore, MD: Human Factors Society.

54 Flecchia, M.A. and Bergeron, R.D. (1987). Specifying complex dialogues in ALGAE. In CHI + GI 1987 Conference Proceedings (pp.229-234). Toronto, Canada: ACM/SIGCHI and ACM/SIGGRAPH.

55 Foley, J., Gibbs, C., Kim, W.C. and Kovacevic, S. (1988). A knowledge-based user interface management system. CHI'88 Conference Proceedings (pp.67-72). Washington, DC: ACM/SIGCHI.

56 Foltz, P.W., Davies, S.E. and Polson, P.G. (1988). Transfer between menu systems. CHI'88 Conference Proceedings (pp.107-112). Washington, DC: ACM/SIGCHI.

57 Fotta, M.E. (1986). Applications of artificial intelligence to improving the intention stage of the human-computer interaction. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.175-178). Dayton, O: Human Factors Society.

58 Francik, E.P. and Kane, R.M. (1987). Optimizing visual search and cursor movement in pull-down menus. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.722-726). New York, NY: Human Factors Society.

59 Furnas, G.W., Landauer, T.K. and Gomez, L.M. (1987). The vocabulary problem in human-system communications. Communications of the ACM, 30(11), 964-971.

60 Gaines, B.R. and Shaw, M.L.G. (1986). From timesharing to the sixth generation: the development of human-computer interaction. International Journal of Man-Machine Studies, 24(1), 1-28.

61 Gaines, B.R. and Shaw, M.L. (1986). Foundations of dialogue engineering: the development of human-computer interaction II. International Journal of Man-Machine Studies, 24(2), 101-123.

62 Galitz, W.O. (1985). Handbook of screen format design (Revised edition). Wellesley, MA: QED Information Sciences, Inc.

63 Gaver, W.W. (1986). Auditory icons: using sound in computer interfaces. Human Computer Interaction, 2(2), 167-177.

64 Gittens, D. (1986). Icon-based human-computer interaction. International Journal of Man-Machine Studies, 24(6), 519-544.

65 Good, M. (1988). User interface consistency in the DECwindows program. Proceedings of the Human Factors Society 32nd Annual Meeting (pp.259-263). Anaheim, CA: Human Factors Society.

66 Gould, J.D. (1988). How to design usable systems. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

67 Gould, J.D., Boies, S.J., Meluson, M., Rasamny, M. and Vosburgh, A.M. (1988). Empirical evaluation of entry and selection methods for specifying dates. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.279-283). Anaheim, CA: Human Factors Society.

68 Gould, J. D. and Lewis, C. (1985). Designing for usability: Key principles and what designers think. In Shackel, B. (Ed.), INTERACT '84: First conference on human-computer interaction. Amsterdam: North-Holland.

69 Gray, J. (1986). The role of menu titles as a navigational aid in hierarchical menus. SIGCHI bulletin, 17(3), 33-40.

70 Green, P. and Wei-Haas, L. (1985). The rapid development of user interfaces: experience with the Wizard of Oz method. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.470-474). Baltimore, MD: Human Factors Society.

71 Greene, S.L., Gould, J.D., Boies, S.J., Meluson, A. and Rasamny, M. (1988). Entry-based versus selection-based interaction methods. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.284-287). Anaheim, CA: Human Factors Society.

72 Green, T. and Demonds, E. (Eds) (1985). The ergonomics of the user interface. Philadelphia: Taylor & Francis.

73 Green, T.R.G. and Payne, S.J. (1984). Organization and learnability in computer languages. International Journal of Man-Machine Studies. 21, 7-18.

74 Grudin, J. and Barnard, P. (1984). The cognitive demands of learning and representing command names for text editing. Human Factors, 26(4), 407-422.

75 Grudin, J. and Barnard, P. (1985). When does an abbreviation become a word? And related questions. In CHI'85 Conference Proceedings (pp.121-126). San Francisco, CA: ACM/SIGCHI.

76 Guida, G. and Tass, C. (1983). IR-NLI: (Information Retrieval Natural Language Interface) an expert natural language interface to online data bases. In Proceedings of the Conference on Applied Natural Language Processing, Santa Monica, California. (AD-P001 164/3)

77 Halpin, S.M. and McCollough, G.W. (1986). User-computer interaction. In Zeidner, J. (Ed), Human Productivity Enhancement Training and Human Factors in System Design. Volume I. New York, Praeger.

78 Hammond, N.V., Barnard, P.J., Morton, J. and Long, J.B. (1987). Characterizing user performance in command-driven dialogue. Behaviour & Information Technology, 6(2), 159-205.

79 Hansen, G.W. and Hansen, J.V. (1987). Procedural and non-procedural query languages revisited: a comparison of relational algebra and relational calculus. International Journal of Man-Machine Studies, 26(6), 683-694.

80 Harker, S. (1987). Rapid prototyping as a tool for user centered design. In Salvendy, G. (Ed). Proceedings of the Second International Conference on Human-Computer Interaction, Honolulu, Hawaii.

81 Harpster, J.L. (1987). Random vs ordered menus in self-terminating menu searches. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.718-7210). New York, NY: Human Factors Society.

82 Hartson, H. R., Roach, J. W., Callan, J. E. III, Nickson, M. M., Pittman, J. A. and Yunten, T. (1986). Dialogue management as part of software engineering methodology. In Ehrich, R. E. and Williges, R. C. Human-Computer Dialogue Design. New York: Elsevier.

83 Hauptmann, A.G. and Green, B.F. (1983). A comparison of command, menu-selection and natural language computer programs. Behaviour and Information Technology, 2(2), 163-178.

84 Hauptmann, A.G. and Rudnicky, A.I. (1988). Talking to computers: an empirical investigation. International Journal of Man-Machine Studies, 28(6), 583-604.

85 Hayes, P. J. (1985). Executable interface definitions using form-based interface abstractions. In Hartson, H. R. Advances in Human-Computer Interaction. Norwood, NJ: Ablex Publishing Corp.

86 Hayes, B.C. and Williges, R.C. (1986). Defining search strategies in information retrieval. Proceedings of the 1986 IEEE International Conference on Systems, Man and Cybernetics. New York: IEEE, pp. 1092-1096.

87 Hefley, W. (1988). Designing real-time, decision support computer-human interaction. SIGCHI bulletin, 20(2), 67-69.

88 Hendrickson, J.J. and Williams, R.D. (1988). Effect of input device on user performance with a menu-based natural language interface (NOSC-TD-1224). San Diego, CA: Naval Ocean Systems Center. (AD-A193 095/7/XAB)

89 Henno, J. (1988). User-friendly syntax design and presentation. International Journal of Man-Machine Studies, 28(5), 551-572.

90 Hewett, T.T. and Meadow, C.T. (1986). On designing for usability: an application of four key principles. In CHI'86 Conference Proceedings (pp.247-252). Boston, MA: ACM/SIGCHI.

91 Hoffman, S.C. (1986). Designing user interfaces: a classroom experience in iterative software design. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.1366-1370). Dayton, O: Human Factors Society.

92 Hopkin, V.D. (1984). Some human-factors implications of expert systems. Behavior and Information Technology, 3(1), 79-83.

93 Hill, R. (1987). Even-response systems--a technique for specifying multi-threaded dialogues. In CHI + GI 1987 Conference Proceedings (pp.241-2 18). Toronto, Canada: ACM/SIGCHI and ACM/SIGGRAPH.

94 Hix, D.H. (1986). Assessment of an interactive environment for developing human-computer interfaces. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.1349-1353). Dayton, O: Human Factors Society.

95 Hix, D.H. and Hartson, H.R. (1986). An interactive environment for dialogue development: its design, use and evaluation; or, is Aide useful? In CHI'86 Conference Proceedings (pp.228-234). Boston, MA: ACM/SIGCHI.

96 Hollan, J.D., Hutchins, E.L. and Weitzman, L. (1984). STEAMER: an interactive inspectable simulation-based training system. AI Magazine, 15-27.

97 Hollands, J.G. and Merikle, P.M. (1987). Menu organization and user expertise in information search tasks. Human Factors, 29(5), 577-586.

98 Hodgson, G.M. and Ruth, S.R. (1985). The use of menus in the design of on-line systems: a retrospective view. SIGCHI bulletin, 17(1), 16-23.

99 Hoyos, C. G., Gestalter, H., Strube, V. and Zang, G. (1987). Software design with the rapid prototyping approach: A survey and some empirical results. In Salvendy, G. (Ed). Proceedings of the Second International Conference on Human-Computer Interaction, Honolulu, Hawaii.

100 Hurd, J.C. (1985). Standardizing the software user interface. Human Factors Society Bulletin, 28(9), 3-4.

101 Hutchins, E.L., Hollan, J.D. and Norman, D.A. (1985). Direct manipulation interfaces. Human Computer Interaction, 1(4), 311-338.

102 Jacob, R.J.K. (1986). An executable specification technique for describing human-computer interaction. In Hartson, H. R. Advances in Human-Computer Interaction. Norwood, NJ: Ablex Publishing Corp.

103 Jamar, P.G. (1986). Lost in computerland: functional model-enhanced training and interface designs. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.497-501). Dayton, O: Human Factors Society.

104 Jarke, M., Turner, J.A., Stohr, E.A., Vassiliou, Y., White, N.H. and Michielsen, K. (1985). A field evaluation of natural language for data retrieval. IEEE Transactions on Software Engineering, SE-11(1), 97-113

105 Jeffries, R. and Rosenberg, J. (1987). Comparing a form-based and a language-based user interface for instructing a mail program. In CHI + GI 1987 Conference Proceedings (pp.261-266). Toronto, Canada: ACM/SIGCHI and ACM/SIGGRAPH.

106 Johnson, D.H. and Hartson, H.R. (1983). Issues in interaction language specification and representation (CSIE-83-15). Blacksburg, VA: Virginia Polytechnic Inst. and State Univ., Computer Science, Industrial Engineering and Operations Research. (AD-A137 478/4)

107 Karat, J. (1987). Evaluating user interface complexity. In Proceedings of the Human Factors Society 31st Annual Meeting. (pp.566-570). New York, NY: Human Factors Society

108 Katzeff, C. (1986). Dealing with a database query language in a new situation. International Journal of Man-Machine Studies, 25(1), 1-18.

109 Kay, D.S. and Black, J.B. (1984). Changes in knowledge representations of computer systems with experience. In Proceedings of the Human Factors Society 28th Annual Meeting (pp.963-967). San Antonio, TX: Human Factors Society.

110 Karat, J., McDonald, J.E. and Anderson, M. (1986). A comparison of menu selection techniques: touch panel, mouse and keyboard. International Journal of Man-Machine Studies, 25(1), 73-88.

111 Kato, T. (1986). What "question-asking protocols" can say about the user interface. International Journal of Man-Machine Studies, 25(6), 659-674.

112 Kieras, D. and Polson, P.G. (1985). An approach to the formal analysis of user complexity. International Journal of Man-Machine Studies, 22, 365-394.

113 Kigler, J.I. (1984). The depth/breadth trade-off in the design of menu-driven user interfaces. International Journal of Man-Machine Studies, 20,201-213.

114 Kiesler, S., Zubrow, D., Moses, A.M. and Geller, V. (1985). Affect in computer-mediated communication: an experiment in synchronous terminal-to-terminal discussion. Human Computer Interaction, 1(1), 77-104.

115 Kuhn, A.D. and Bixby, R.L. (1987). DoD Gateway Information System (DGIS): common command language: the first prototyping and the decision for artificial intelligence (DTIC/TR-87/19). Alexandria, VA: Office of Information Systems and Technology. (AD-A185 950/3/XAB)

116 Koved, L. and Shneiderman, B. (1986). Embedded menus: menu selection in context. Communications of the ACM, 29, 312-318.

117 Landauer, T.K. and Nachbar, D.W. (1985). Selection from alphabetic and numeric menu trees using a touch screen: breadth, depth, and width. In CHI'85 Conference Proceedings (pp.73-78). San Francisco, CA: ACM/SIGCHI.

118 Landee-Thompson, S. (1988). User-computer interface technology: an assessment of the current state of the art. Ft. Leavenworth, KS: U.S. Army Research Institute for the Behavioral and Social Sciences. (AD-A196 839)

119 Laversonh, A., Norman, K. and Shneiderman, B. (1987). An evaluation of jump-ahead techniques for frequent menu users. Behavior and Information Technology, 6, 97-108.

120 Lee, E., Whalen, T., McEwen, S. and Latremouille, S. (1984). Optimizing the design of menu pages for information retrieval. Ergonomics, 27(10), 1051-1070.

121 Lee, E. and MacGregor, J. (1985). Minimizing user search time in menu-retrieval systems. Human Factors, 27(2), 157-162.

122 Lehner, P.E. and Zirk, D.A. (1987). Cognitive factors in user/expert-system interaction. Human Factors, 29(1), 97-110.

123 Lindgaard, G. and Perry, L. (1988). Making life easier for computer novices: Some factors determining initial performance. Ergonomics, 31(5), 803-816.

124 Lorenzetti, R., Williamson, J., Hoffman, L., Beck, T. and Maguire, F. (1988). A case study in large scale computer human interaction design. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.1041-1045). Anaheim, CA: Human Factors Society.

125 Lowgren, J.E. (1987). Are we boxing ourselves in with the UIMS box? In Proceedings of the Human Factors Society 31st Annual Meeting (pp.22-24). New York, NY: Human Factors Society.

126 Lowgren, J. (1988). History, state and future of User Interface Management Systems. SIGCHI bulletin, 20(1), 32-44.

127 Maeda, K., Miyake, Y., Nievergelt, J. and Saito, Y. (1984). A comparative study of man-machine interfaces in interactive systems. SIGCHI bulletin, 16(2), 44-61.

128 MacGregor, J., Lee, E. and Lam, N. (1986). Optimizing the structure of database menu indexes: a decision model of menu search. Human Factors, 28(4), 387-400.

129 MacGregor, J. and Lee, E. (1987). Menu search: random or systematic? International Journal of Man-Machine Studies, 26(5), 627-632.

130 Margono, S. and Shneiderman, B. (1987). A study of file manipulation by novices using commands vs. direct manipulation. Proceedings of the 26th Annual Technical Symposium. Washington, DC: ACM.

131 Mark, W. (1985). Knowledge-based user interface design. Human Computer Interaction, 1(4), 339-359.

132 Mark, W. (1986). Knowledge-based interface design. In Norman, D. A. and Draper, S. W. User Centered System Design. Hillsdale, NJ: Lawrence Erlbaum Assoc.

133 Martin, J. (1973). Design of man-computer dialogues. Englewood Cliffs, NJ: Prentice-Hall.

134 Martin, M.P. and Fuerst, W.L. (1987). Using computer knowledge in the design of interactive systems. International Journal of Man-Machine Studies, 26(3), 333-342.

135 Mason, J.A. and Edwards, J.L. (1988). Surveying projects on intelligent dialogue. International Journal of Man-Machine Studies, 28(2&3), 259-308.

136 Mason, M.V. (1986). Adaptive command prompting in an on-line documentation system. International Journal of Man-Machine Studies, 25(1), 33-52.

137 McDonald, J.E., Dearholt, D.W., Paap, K.R., Schvaneveldt, R.W. (1986). A formal interface design methodology based on user knowledge. In CHI'86 Conference Proceedings (pp.291-297). Boston, MA: ACM/SIGCHI.

138 McDonald, J.E., Dayton, T. and Mcdonald, D.R. (1988). Adapting menu layout to tasks. International Journal of Man-Machine Studies, 28(4), 417-437.

139 McDonald, J.E., Molander, M.E. and Noel, R.W. (1988). Color-coding categories in menus. CHI'88 Conference Proceedings (pp.101-106). Washington, DC: ACM/SIGCHI.

140 Meister, D. and Sullivan, D. J. (1967). A further study of the use of human factors information by designers. Canoga Park, CA: The Bunker-Ramo Corp.

141 Meister, D., Sullivan, D. J. and Finley, D. L. (1969). The effect of amount and timing of human resources data on subsystem design. Canoga Park, CA: The Bunker-Ramo Corp.

142 Melkus, L.A. and Torres, R.J. (1988). Guidelines for the use of a prototype in user interface design. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.370-374). Anaheim, CA: Human Factors Society.

143 Mills, C.B. (1987). Usability testing in the real world. SIGCHI bulletin, 19(1), 43-46.

144 Monk, A. (1985). Fundamentals of human-computer interaction. San Diego, CA: Academic Press.

145 Mosier, J.N. and Smith, S.L. (1985). Application of guidelines for designing user interface software. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.946-949). Baltimore, MD: Human Factors Society.

146 Mueller, M.J. (1988). Multifunctional cursor for direct manipulation user interfaces. CHI'88 Conference Proceedings (pp.89-94). Washington, DC: ACM/SIGCHI.

147 Nielsen, J. (1986). A virtual protocol model for computer-human interaction. International Journal of Man-Machine Studies, 24(3), 301-312.

148 Norman, D.A. (1983). Design rules based on analyses of human error. Communications of the ACM, 26(4), 254-258.

149 Norman, D. A. and Draper, S. W. (1986). User centered system design: new perspectives on human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Assoc.

150 Nosek, J.T. and Ahrens, J.D. (1986). An experiment to test user validation of requirements: data-flow diagrams vs task-oriented menus. International Journal of Man-Machine Studies, 25(6), 675-684.

151 Ogden, W. and Kaplan, C. (1986). The use of And and Or in a natural language computer interface. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.829-833). Dayton, O: Human Factors Society.

152 Ogden, W.C. (1988). Using natural language interfaces. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

153 Overmyer, S.P. (1987). Rapid prototyping in the design of large-scale interactive information systems. Human Factors Society Bulletin, 30, 3-4.

154 Paap, K.R. and Roske-Hofstrand, R.J. (1986). The optimal number of menu options per panel. Human Factors, 28(4), 377-386.

155 Paap, K.R. and Roske-Hofstrand, R.J. (1988). Design of menus. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

156 Parkinson, S.R., Hill, M.D., Sisson, N. and Viera, C. (1988). Effects of breadth, depth and number of responses on computer menu search performance. International Journal of Man-Machine Studies, 28(6), 683-692.

157 Parton, D., Huffman, K., Pridgen, P, Norman, K. and Shneiderman, B. (1985). Learning a menu selection tree: training methods compared. Behavior and Information Technology, 4, 81-91.

158 Payne, S.J. and Green, T.R.G. (1986). Task-action grammars: a model of the mental representation of task languages. Human Computer Interaction, 2(2), 93-134.

159 Penn, D. (1987). User interface design tools. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.25-29). New York, NY: Human Factors Society.

160 Perlman, G. (1984). Making the right choices with menus. In Shackel, B. (Ed.), INTERACT '84: First conference on human-computer interaction (pp.291-295). Amsterdam: North-Holland.

161 Peterson, R.J., Hester, T. and Yorchak, J.P. (1984). EXAMINE: an expert system to mediate human-computer dialogues. In Proceedings of the Human Factors Society 28th Annual Meeting (pp.889-893). San Antonio, TX: Human Factors Society.

162 Pew, R.W. (1988). Human factors issues in expert systems. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

163 Pfaff, G.R. (1985). User interface management systems. Berlin: Springer-Verlag.

164 Phillips, M.D., Bashinski, H.S., Ammerman, H.L. and Fligg, C.M. (1988). A task analytic approach to dialogue design. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

165 Quinn, L. and Russel, D.M. (1986). Intelligent interfaces: user models and planners. In CHI'86 Conference Proceedings (pp.314-320). Boston, MA: ACM/SIGCHI.

166 Rasmussen, J. (1986). Information procession and human-machine interaction: An approach to cognitive engineering. New York, NY: North-Holland.

167 Reisner, P. (1988). Query languages. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

168 Reitman-Olson, J., Carroll, J.M. and Anderson, N. (Eds.) (1987). What does the user of software know? Mental models and software human factors. Washington, DC: National Academy Press.

169 Rich, E. (1984). Natural-language interfaces. Computer, 7(9), 39-47.

170 Richards, J.N.J., Bez, H.E., Gittins, D.T. and Cookde, D.J. (1986). On methods for interface specification and design. International Journal of Man-Machine Studies, 24(6), 545-568.

171 Rogers, W.H. and Moeller, G. (1984). Comparison of abbreviation methods: measures of preference and decoding performance. Human Factors, 26(1), 49-60.

172 Roske-Hofstrand, R.J. and Paap, K.R. (1986). Cognitive networks as a guide to menu organization: an application in the automated cockpit. Ergonomics, 29(11), 1301-1312.

173 Rouse, W.B., Geddes, N.D. and Curry, R.E. (1987). An architecture for intelligent interfaces: outline of an approach to supporting operators of complex systems. Human-Computer Interaction, 3(2), 87-122.

174 Roth, E.M., Bennett, K.B. and Woods, D.D. (1987). Human interaction with an "intelligent" machine. International Journal of Man-Machine Studies, 27(5&6), 479-526.

175 Runciman, C. and Thimbleby, H. (1986). Equal opportunity interactive systems. International Journal of Man-Machine Studies, 25(4), 439-452.

176 Rushinek, A. and Rushinek, S.F. (1986). What makes users happy? Communications of the ACM, 29, 594-598.

177 Saga, A.D. (1985). The cognitive model: an approach to designing the human-computer interface. SIGCHI bulletin, 16(3), 36-40.

178 Sage, A. P. (Ed.) (1987). System design for human interaction. New York, NY: IEEE Press.

179Savage, R.E. and Habinek, J.K. (1984). A multilevel menu-driven user interface: Design and evaluation through simulation. In Thomas, J.C. and Schneider, M.L. (Eds.) Human factors in computer systems. Norwood, NJ: Ablex.

180 Schvaneveldt, R.W., Durso, F.T., Goldsmith, T.E., Breen, T.J., Cooke, N.M., Tucker, R.G. and DeMaio, J.C. (1985). Measuring the structure of expertise. *International Journal of Man-Machine Studies*, 23, 699-728.

181 Schwartz, J.P. and Norman, K.L. (1986). The importance of item distinctiveness on performance using a menu selection system. Behaviour and Information Technology, 5, 173-182.

182 Scott, M.L. and Yap, S. (1988). A grammar-based approach to the automatic generation of user-interface dialogues. CHI'88 Conference Proceedings (pp.73-78). Washington, DC: ACM/SIGCHI.

183 Seppala, P. and Salvendy, G. (1985). Impact of depth of menu hierarchy on performance effectiveness in a supervisory task: computerized flexible manufacturing system. Human Factors, 27(6), 713-722.

184 Shackel, B. (1984). Information technology - a challenge to ergonomics and design. Behavior and Information Technology, 3, 263-275.

185 Shaw, B. E. and McCauley, M. E. (1985). Person computer dialogue: A human engineering data base supplement (AFAMRL-TR-85-013). Wright-Patterson AFB, O: Air Force Aerospace Medical Research Laboratory.

186 Shaw, M. (1986). An input-output model of interactive systems. In CHI'86 Conference Proceedings (pp.261-273). Boston, MA: ACM/SIGCHI.

187 Shinar, D., Stern, H.I., Bubis, G. and Ingram, D. (1985). The relative effectiveness of alternative selection strategies in menu driven computer programs. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.645-650). Baltimore, MD: Human Factors Society.

188 Shinar, D. and Stern, H.I. (1987). Alternative option selection methods in menu-driven computer programs. Human Factors, 29(4), 453-460.

189 Shneiderman, B. (1983). Direct manipulation: a step beyond programming languages. IEEE Computer, 16(8), 57-69.

190 Shneiderman, B. (1986). Designing the User Interface: Strategies for effective human-computer interaction. Menlo Park, CA: Addison-Wesley.

191 Shneiderman, B. (1988). We can design better user interfaces: A review of human-computer interaction styles. Ergonomics, 31(5), 699-710.

192 Shurtleff, M.S., Jenkins, J.A., and Sams, M.R. (1988). Deriving menu structures through modal block clustering: a promising alternative to hierarchical techniques. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.347-351). Anaheim, CA: Human Factors Society.

193 Sidorsky, R. C., Parrish, R. N., Gates, J. L. and Munger, S. J. (1983). Design guidelines for user transactions with battlefield automated systems: Prototype for a handbook. Alexandria, VA: U. S. Army Research Institute for the Behavioral and Social Sciences. (AD-A153 231)

194 Simes, D. K. and Sirsky, P. A. (1985). Human Factors: An exploration of the psychology of human-computer dialogues. In Hartson, H. R. (Ed.) Advances in human-computer interaction (Vol. 1). Norwood, NJ: Ablex Publishing Corp.

195 Simmons, R.F. (1986). Man-machine interfaces: can they guess what you want? IEEE Expert, 1(1), 86-94.

196 Singley, M.K. and Anderson, J.R. (1987). A keystroke analysis of learning and transfer in text editing. Human Computer Interaction, 3(3), 223-274.

197 Sisson, N., Parkinson, S.R. and Snowberry, K. (1986). Considerations of menus structure and communication rate for the design of computer menu displays. International Journal of Man-Machine Studies, 25(5), 479-490.

198 Small, D. and Weldon, L. (1983). An experimental comparison of natural and structured query languages. Human Factors, 25, 253-263.

199 Smith, E. and Siochi, A. (1988). Software usability: requirements by evaluation. Proceedings of the Human Factors Society 32nd Annual Meeting (pp.264-266). Anaheim, CA: Human Factors Society.

200 Smith, R.G., Lafue, G.M.E., Schoen, E. and Vestal, S.C. (1984). Declarative task description as a user-interface structuring mechanism. Computer, 17(9), 29-38.

201 Smith, S. L. and Mosier, J. N. (1984). Design guidelines for user-system interface software (ESD-TR-84-190). Hanscom AFB, MA: Electronic Systems Division, Air Force Systems Command. (AD-A154 907).

202 Snowberry, K., Parkinson, S.R. and Sisson, N. (1983). Computer display menus. Ergonomics, 26(7), 699-712.

203 Snowberry, K., Parkinson, S.R. and Sisson, N. (1985). Effects of help fields on navigating through hierarchical menu structures. International Journal of Man-Machine Studies, 22, 479-491.

204 Snyder, K.M., Happ, A.J., Malcus, L., Paap, K.R. and Lewis, J.R. (1985). Using cognitive models to create menus. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.655-658). Baltimore, MD: Human Factors Society.

205 Somberg, B.L. and Dotson, K.E. (1985). Methods of disambiguating command term abbreviations for interactive computer systems. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.659-663). Baltimore, MD: Human Factors Society.

206 Stoddard, M.L. (1985). Development of user interface guidelines for multi-functional, in-house-developed information systems. In Proceedings of the Human Factors Society 29th Annual Meeting (pp.945). Baltimore, MD: Human Factors Society.

207 Taylor, M.M. (1988). Layered protocols for computer-human dialogue. International Journal of Man-Machine Studies, 28(2&3), 175-258.

208 Taylor, R. (1986). Using multiple dialogue modes in a user-system interface to accommodate different levels of user experience: an experimental study (Doctoral thesis). Wright-Patterson AFB, O: Air Force Institute of Technology. (AD-A186 115/2/XAB)

209 Tenhagen, P.J.W. and VanLiere, R. (1987). Model for graphical interaction (Report No. CWI-CS-R8718). Washington, DC: National Aeronautics and Space Administration. (N88-20866/5/XAB)

210 Thomas, J.J. (1983). Architecture for a User Interface Management System. In Workshop on user interface management system, Frankfort, F.R. Germany. Washington, DC: Department of Energy. (DE84003982)

211 Thomas, J. and Schneider, M. (1984). Human factors in computer systems (Volume 3). Norwood, NJ: Ablex.

212 Tijerina, L. (1986). Design guidelines and the human factors of interface design. In Proceedings of the Human Factors Society 30th Annual Meeting (pp.1358-1362). Dayton, O: Human Factors Society.

213 Tullis, T.S. (1985). Designing a menu-based interface to an operating system. In CHI'85 Conference Proceedings (pp.79-84). San Francisco, CA: ACM/SIGCHI.

214 Tullis, T.S. (1988). Screen design. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

215 Van Biljon, W.R. (1988). Extending Petri nets for specifying man-machine dialogues. International Journal of Man-Machine Studies, 28(4), 437-455.

216 Vassiliou, Y. (Ed.) (1984). Human factors and interactive computer systems. Norwood, NJ: Ablex.

217 Verplank, W.L. (1988). Graphic challenges in designing object-oriented user interfaces. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

218 Vicente, K.J. and Rasmussen, J. (1988). On applying the skills, rules, knowledge framework to interface design. Proceedings of the Human Factors Society 32nd Annual Meeting (pp.254-258). Anaheim, CA: Human Factors Society.

219 Wallace, D.F., Anderson, N.S. and Shneiderman, B. (1987). Time stress effects on two menu selection systems. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.727-731). New York, NY: Human Factors Society.

220 Wasserman, A.I. and Shewmake, D.T. (1985). The role of prototypes in the user software engineering (USE) methodology. In Hartson, H.R. (Ed.) Advances in human-computer interaction, 1, (pp. 191-209). Norwood, NJ: Ablex.

221 Whiteside, J., Bennett, J. and Holtzblatt, K. (1988). Usability engineering: our experience and evolution. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

222 Whiteside, J., Jones, S., Levy, P.S. and Wixon, D. (1985). User performance with command, menu and iconic interfaces. In CHI'85 Conference Proceedings (pp.185-191). San Francisco, CA: ACM/SIGCHI.

223 Williams, J.R. (1988). The effects of case and spacing on menu option search time. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.341-343). Anaheim, CA: Human Factors Society.

224 Williges, R. C. and Hartson, H. R. (1986). Human-computer dialogue design and research issues. In Ehrich, R. W. and Williges, R. C. Human-computer dialogue design. New York: Elsevier.

225 Williges, B. H. and Williges, R. C. (1984). Dialogue design considerations for interactive computer systems. In Muckler, F. A. Human Factors Review.

226 Williges, R.C., Williges, B.H. and Elkerton, J. (1987). Software interface design. In Salvendy, G. Handbook of Human Factors. New York: Wiley, pp.1416-1449.

227 Wilson, D.L. and Kuperman, G.G. (1988). Artificial intelligence (AI) system interface attributes: survey and analyses. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.1036-1040). Anaheim, CA: Human Factors Society.

228 Winograd, T. (1987). A language/action perspective on the design of cooperative work. Human Computer Interaction, 3(1), 3-30.

229 Winograd, T. and Flores, F. (1986). Understanding Computers and Cognition. Norwood, NJ: Ablex.

230 Wixon, D. and Good, M. (1987). Interface style and eclecticism: moving beyond categorical approaches. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.571-575). New York, NY: Human Factors Society.

231 Wolf, C.G. and Rhyne, J.R. (1987). A taxonomic approach to understanding direct manipulation. In Proceedings of the Human Factors Society 31st Annual Meeting (pp.576-580). New York, NY: Human Factors Society.

232 Wolf, C.G. (1988). A comparative study of gestural and keyboard interfaces. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp.273-277). Anaheim, CA: Human Factors Society.

233 Yang, Y. (1988). Undo support models. International Journal of Man-Machine Studies, 28(5), 457-482.

234 Zachary, W. (1986). A cognitively based functional taxonomy of decision support techniques. Human Computer Interaction, 2(1), 25-64.

235 Zachary, W. (1988). Decision support systems: designing to extend the cognitive limits. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

236 Ziegler, J.E. and Fahnrich, K.P. (1988). Direct manipulation. In Helander, M. (Ed.) Handbook of Human-Computer Interaction. Amsterdam: North-Holland.

237 Zoeppritz, M. (1986). A framework for investigating language-mediated interaction with machines. International Journal of Man-Machine Studies, 25(3), 295-316.

238 Zoltan-Ford, E. (1984). Reducing variability in natural-language ,interactions with computers. In Proceedings of the Human Factors Society 28th Annual Meeting (pp.768-772). San Antonio, TX: Human Factors Society.

INDEX

[1] page number to text.

Index (cont'd.)

Index (cont'd.)